COMBINING QUALITY OF SERVICE AND TOPOLOGY CONTROL IN DIRECTIONAL HYBRID WIRELESS NETWORKS

THESIS

Michael C. Erwin, Second Lieutenant, USAF

AFIT/GOR/ENS/06-07

DEPARTMENT OF THE AIR FORCE
AIR UNIVERSITY

AIR FORCE INSTITUTE OF TECHNOLOGY

Wright-Patterson Air Force Base, Ohio

AFIT/GOR/ENS/06-07

COMBINING QUALITY OF SERVICE AND TOPOLOGY CONTROL IN DIRECTIONAL HYBRID WIRELESS NETWORKS

THESIS

Presented to the Faculty

Department of Operational Sciences

Graduate School of Engineering and Management

Air Force Institute of Technology

Air University

Air Education and Training Command

In Partial Fulfillment of the Requirements for the

Degree of Master of Science in Operations Research

Michael C. Erwin, BS

Second Lieutenant, USAF

March 2006

AFIT/GOR/ENS/06-07

COMBINING QUALITY OF SERVICE AND TOPOLOGY CONTROL IN
DIRECTIONAL HYBRID WIRELESS NETWORKS

Michael C. Erwin, BS
Second Lieutenant, USAF

Approved:

Raymond W. Staats, Lt Col, USAF (Chairman)	date
Scott R. Graham, Maj, USAF (Member)	date
Dr. Kenneth M. Hopkinson (Member)	date

AFIT/GOR/ENS/06-07

Abstract

Recent advancements in information and communications technology are changing the information environment in both quantitative and qualitative measures. The developments in directional wireless capabilities necessitate the ability to model these new capabilities, especially in dynamic environments typical of military combat operations. This thesis establishes a foundation for the definition and consideration of the unique network characteristics and requirements introduced by this novel instance of the Network Design Problem (NDP). Developed are a Mixed-Integer Linear Program (MILP) formulation and two heuristic strategies for solving the NDP. A third solution strategy using the MILP formulation with a degree-constrained Minimum Spanning Tree starting solution is also considered. The performance of the various methods are evaluated on the basis of solution quality, computation time, and other network metrics via randomly generated data sets for several different problem sizes.

AFIT/GOR/ENS/06-07

To my beautiful wife and daughter.

Acknowledgments

First, I must thank my beautiful loving wife for all of her undying love and support throughout my entire time here at AFIT. I would like to express my sincere appreciation to my faculty advisor, Lt Col Raymond Staats, for his guidance and support throughout the course of this thesis effort. The insight and experience was certainly appreciated. I would also like to thank my committee members Maj Scott Graham and Dr Kenneth Hopkinson for their enthusiastic guidance and support throughout this process.

Michael C. Erwin

Table of Contents

List of Figures

List of Tables

COMBINING QUALITY OF SERVICE AND TOPOLOGY CONTROL IN DIRECTIONAL HYBRID WIRELESS NETWORKS

I. Introduction

Background

Joint Vision (*JV*) *2020*, a Department of Defense document published in June 2000, provides guidance for the continuing transformation of America's military forces [1]. Its purpose is to outline goals that will provide for the formation of a joint military force that is dominant in every phase of military operations. According to *JV 2020*, information superiority is at the core of every military operation. Recent and future advancements in information and communications technology are changing the information environment in both quantitative and qualitative measures. Utilizing these enhanced capabilities and other innovations will require adjustments and alterations in the conduct of military operations. *JV 2020* discusses the need to develop the concept of a global information grid (GIG) that will provide the network-centric environment necessary to achieve the goal of integrating traditional forms of information operations with new ones and multiple sources of information into a fully synchronized information campaign. Among other goals, the GIG aims to fully connect and provide information on demand to warfighters, policy makers, and support personnel. This type of network capability will greatly enhance the combat power of the United States military.

One of the most intriguing enhancements in the advancement of communications technology is the improved utility and availability of wireless communications. Most people are familiar with various forms of wireless communication, whether it is connecting a laptop computer to a network, listening to one's favorite radio station, or text messaging a friend with a cellular phone. Without wireless networks, communication in many mobile environments would be impossible. Mobile environments typical of military combat require the flexibility made possible through wireless communications. However, increasing bandwidth requirements present a challenge to traditional omni-directional communication. The combination of high-bandwidth directional links with directional broadcast radio frequency (RF) links, which together form high bandwidth capabilities for hybrid communication networks, suggest the potential for global connectivity while avoiding some of the fundamental scaling limits associated with omni-directional wireless links [2]. This type of hybrid communication network provides highly desired levels of availability at a relatively low cost to many users. The Internet provides a useful model for the architecture of this type of network, but it does not address the complex demands of a mobile military network.

Definition of Terms

A network is comprised of communicating systems called nodes. Nodes are connected by communication links. The current set of active links form the topology of the network. Directional links are formed when two directional transceivers point toward each other. Directional links are assumed to connect only two nodes (as opposed to omni-directional) and represent a limited resource. Therefore, careful consideration must

be made as to which node a directional transceiver should point to. In a mobile network, links can break as nodes are obscured or move out of range. Thus mobile networks have highly dynamic topologies.

Statement of the Problem

Management of the dynamic environment inherent in tactical military operations will benefit greatly from the utilization of hybrid wireless networks. Network users in this sort of environment will have complex demands and preferences. Since there is more than one type of link in a hybrid network, the different types of links may satisfy the user's preferences to different degrees. For example, "network administrators" may determine the most important network characteristics are latency, power utilization, and probability of transmission interception [3]. Also, some users may have a higher or lower priority than other users. At any given time in this environment, the number of users, who they can connect to, how many and what type of links each user can establish, and the users' preferences and priorities may change. The problem lies in determining the optimal network topology, or where links should and should not be established, given the network characteristics and requirements. This research focuses on the development of a method to provide a network topology that satisfies the demands and requirements of its users at a minimum (or near-minimum) cost. Network characteristics that will be considered to produce a topology are the number of nodes, number and type of communication interfaces at each node, commodity and bandwidth requirements, commodity priorities, cost of establishing communication links, and cost of using those links.

Research Approach

Previous research has addressed network flow and routing problems, including network design formulations. The type of hybrid network proposed in the foregoing question does not yet exist, hence this instance of the network design problem is new. Existing methods for solving minimum cost network design problems will be examined for adaptation to this problem. More than one solution technique will be implemented and evaluated. Analyzing the trade-off between computation (time) efficiency and solution quality is important, because information operations in a combat environment are very time sensitive. Other network topology metrics such as delay and satisfaction of bandwidth requirements will also be considered and may provide valuable insight for this problem.

A Minimum-Cost Spanning Tree (MST) algorithm may provide a quick starting solution that guarantees connectivity at a minimum cost. Then other methods can be examined to establish a desired level of redundancy or connectivity, for example. As with most network problems, this one can be formulated as a Mixed Integer Linear Program (MILP). As the network design problem increases in size, however, solving the MILP suffers dramatic losses in computational efficiency. Therefore alternative solution methods will be examined to determine their effectiveness towards this problem.

Scope and Limitations

The different solution strategies will be for a general directional wireless hybrid communications network. Arbitrary network characteristics and requirements as well as user preferences and priorities can be used without loss of generality. The feasibility,

advantages, and disadvantages of each solution strategy will be determined, thus establishing groundwork for future research in the area of hybrid wireless network topology control.

Summary

This chapter outlines the background and motivation for this research. Chapter II reviews the pertinent concepts, methods, and techniques that will be considered for the implementation and evaluation of the solution strategies used in this research. Chapter III describes the methodology used to determine the feasibility, advantages, and disadvantages of each solution strategy used in this research. Chapter IV presents the results of the analysis as well as conclusions about topology control in hybrid wireless networks. Chapter V summarizes the methodology, observations, results, and conclusions of this research, and offers suggestions for future research.

II. Literature Review

Introduction

We now review the literature pertaining to the problem we consider in this research. We begin with a review of communication networks and the network design problem. We then review several instances of the network design problem and techniques that have been applied to solve it. Next we discuss in further detail the mixed-integer linear programming approach and different solution methods and decomposition techniques. We then review common network topology metrics that could also be used in this work to compare the results of different topology design strategies.

Communications Networks

In general, communication systems involve sending information from some source to some destination. These source and destination points are commonly called nodes [17:50]. A communications network is a collection of "stations" that transmit information signals from source nodes to destination nodes via a transmission medium. Pooch, Machuel, and McCahn define telecommunications as "the art and science of communicating at a distance" [15:3]. Telecommunications networks are growing in importance and have become an essential part of life. Indeed, as of 1998, the telecommunications market approached $1 trillion per year [18:1].

A network design is a blueprint for building a network [19]. With so much capital invested in the development and implementation of communications networks, an important question to consider is how to design a network while trying to balance the

associated costs with quality of service (QoS) and benefits provided to network users [11:1].

Network Design Problem

Many researchers have attacked the problem of network design. Interestingly, recent advancements in communications technology and capabilities provide us with seemingly numberless areas of research in the network design problem (NDP). Researchers differ in the definition of the NDP, their suggested solution methods, and even the type of network they consider. Most studies, it seems, are conducted on a particular instance or type of network with, for example, a specific transmission medium and a certain design objective. We limit our review to the literature that is most similar and pertinent to the problem considered in this research.

Ahuja, Magnanti, and Orlin [5] describe the NDP as having the flexibility of designing a network as well as determining its optimal flow, or routing. If an arc is used, a fixed cost is incurred. There is an additional cost for the usage or flow along an arc. The problem, according to Ahuja et al, is to find the design that minimizes the total systems cost, or the sum of the design cost and the routing cost. By defining the problem in this general manner, it can be applied to more than one type of network. Suggested applications include the design of telecommunication or computer networks, load planning in the trucking industry, and design of production schedules [5:627]. Cosares and Rispoli [39] define the NDP in a similar manner. The focus of their study, however, is on a case of the NDP in which all traffic originates at some central location, so the underlying sub-network will form a spanning tree.

Returning to the Ahuja et al. [5] work, they consider the uncapacitated NDP. They route multiple commodities on the network. Each commodity k has a single source node s^k and a single destination node d^k. Once an arc is introduced into the network, there is sufficient capacity to route all of the flow by all commodities on this arc [5:627]. The formulation of the model follows:

Let x^k denote the vector of flows of commodity k on the network.

Let x_{ij}^k denote the fraction of the required flow of commodity k to be routed from the source s^k to the destination d^k that flows on arc (i, j).

Let c^k denote the cost vector for commodity k (c_{ij}^k is the per unit cost for commodity k on arc (i, j) multiplied by the flow requirement of that commodity).

Let f denote the fixed cost vector for the construction of each arc in the network.

Let y_{ij} be a zero-one variable indicating whether arc (i, j) is selected as part of the network design.

$$\text{Minimize} \quad \sum_{1 \le k \le K} c^k x^k + fy \tag{2.1}$$

subject to

$$\sum_{\{j:(i,j)\in A\}} x_{ij}^k - \sum_{\{j:(j,i)\in A\}} x_{ji}^k = \begin{cases} 1 & \text{if } i = s^k \\ -1 & \text{if } i = d^k \\ 0 & \text{otherwise} \end{cases} \qquad \forall\ i \in N, k = 1, 2, \ldots, K \tag{2.2}$$

$$x_{ij}^k \le y_{ij} \qquad \forall\ (i, j) \in A, k = 1, 2, \ldots, K \tag{2.3}$$

$$x_{ij}^k \ge 0 \qquad \forall\ (i, j) \in A,\ k = 1, 2, \ldots, K \tag{2.4}$$

$$y_{ij} \text{ is binary} \qquad \forall\ (i, j) \in A \tag{2.5}$$

In this problem, the "forcing constraints" (2.3) state that if we do not choose arc *(i,j)*, we cannot flow any fraction of commodity *k*'s demand on this arc, and if we select arc *(i,j)*, the flow of commodity *k* on this arc is unlimited. Ahuja et al. note that if the forcing constraints are removed, the resulting model in the flow variables x^k decomposes into a set of independent shortest path problems, one for each commodity *k*. Therefore, Ahuja et al. argue, the model is an ideal candidate for the application of Lagrangian relaxation. This problem can be incredibly large, especially in cases such as communication networks where each node is sending messages to every other node [5:628].

Alevras, Groetschel, Jonas, Paul, and Wessaely [7] investigate the design of a "survivable" telecommunications network. They seek an integrated approach to the NDP, where cost effectiveness, survivability, and network management aspects are taken into account simultaneously to achieve a solution that appears efficient in each aspect. Given a communications demand between each pair of switching nodes in a region and a set of valid capacities, they consider the problem of determining the topology of a telecommunications network connecting the given nodes and with capacities for each link such that the communications demands are satisfied at minimum cost. The solution also includes the routings for each demand.

In large networks, according to Cahn [19], there are nodes that are more important than other nodes. For example, nodes that are close to numerous other nodes might have more traffic. If the division between important and less important nodes is distinct, deciding which nodes should be in the backbone is somewhat obvious. This concept of division of the nodes leads to a division of the design problem into two pieces – the

access design, which gets the traffic from the small sites to the backbone, and the backbone design, which builds a "mesh" between the large nodes. According to Cahn, it is impossible to say whether backbone design or access design comes first, whereas both designs affect each other [19].

Most researchers who take the access network design approach tend to start with the backbone. Premkumar and Chu [11] explain how network design can be considered in terms of two broad areas – backbone network design and local access network design. The primary focus of their study is the design of a backbone network to satisfy given design and reliability criteria. They describe the most common topology design approach, which is the minimum spanning tree (MST) problem, which attempts to find a spanning tree that connects all the nodes of the network at a minimum cost [11:1]. The cost associated with each link is general and can be based, for example, on distance, capacity, or the quality of the link. Premkumar and Chu use a variation of the MST problem called the degree-constrained MST (dcMST) problem. The dcMST problem reflects the constraints in real world design in terms of the number of links connecting to each node [11:2].

Guéret, Prins, and Sevaux [51:182] formulate the MST problem as an integer program (IP) to solve a simple telecommunications NDP. Their IP formulation follows.

Let $N = \{1, 2, ..., n_N\}$ be the set of all n_N nodes in the network.

Let A be the $(n_N \times n_N)$ node-incidence matrix with $a_{ij} = 1$ if node i is incident to node j, and $a_{ij} = 0$ otherwise.

Let y_{ij} denote the binary decision variable indicating whether or not edge (i, j) is chosen. $y_{ij} = 1$ if chosen, $y_{ij} = 0$ otherwise.

Let c_{ij} denote the cost of including edge (i, j) in the network.

Let $Level_i$ denote the integer value that corresponds to the number of links in the path from the root node to node *i*.

$$\text{Minimize } z = \sum_{\{(i,j):\, a_{ij}=1\}} c_{ij} y_{ij} \tag{2.6}$$

subject to

$$\sum_{\{(i,j):\, a_{ij}=1\}} y_{ij} = (n_N - 1) \tag{2.7}$$

$$Level_j \geq Level_i + 1 - n_N + n_N y_{ij} \qquad \forall i, j \in N \tag{2.8}$$

$$\sum_{\{j:\, a_{ij}=1\}} y_{ij} = 1 \qquad \forall i \in \{2, 3, \ldots, n_N\} \tag{2.9}$$

$$y_{ij} \text{ is binary} \qquad \forall i, j \in N \tag{2.10}$$

$$Level_i \geq 0 \text{ is integer} \qquad \forall i \in N \tag{2.11}$$

Constraint (2.7) ensures that only $(n_N - 1)$ edges are selected – the number of edges in a spanning tree of a graph with n_N nodes. Equations (2.8) and (2.9) are sub-cycle constraints. Guéret, et al. consider the tree with its directed edges departing from a root node. Every node is then assigned a level value $Level_i$ that can be interpreted as the length (in terms of number of links) of the path connecting node *i* to the root node. For example, $Level_i$ for the root node is 0, and $Level_i$ for any node connected to the root node is 1. Equation (2.8) only detects cycles with links directed around the cycle. Consequently we add equation (2.9) to prevent all cycles. Node 1 is arbitrarily chosen as the root node. Every node must be connected to at least one other node. Since a tree does not contain cycles there must be a single path from every node in the tree to the root

node, which means that every node *i* other than the root node has exactly one outgoing link [51:183-184].

The complexity of the MST problem increases significantly as the number of nodes increases, making it impractical to use traditional mathematical models to solve problems with a large number of nodes [11:2]. Many heuristic solutions have been developed to solve large MST problems. The well-known forerunners in this field are by Kruskal and Prim [23,24], who both developed algorithms to solve large MST problems. Researchers have since modified these heuristic algorithms to solve the dcMST problem. Narula and Ho [25] proposed three heuristic algorithms – primal, dual, and branch and bound. Savelsbergh and Volgenant [26] introduced an “edge exchange” algorithm, which was found to perform better than the primal or dual algorithms. Yet another heuristic approach has been the use of genetic algorithms (GA). Genetic algorithms have been effectively used to solve combinatorial optimization problems in telecommunications design [27, 28, and 29]. Only recently, however, have researchers examined the use of GA for solving the dcMST problem [11, 30, 31, and 32]. Premkumar and Chu found that in the context of the dcMST problem, GA methods provide better solution quality than commonly used heuristics with a difference of about twenty percent. GA methods, however, require far more computational time (three to four orders of magnitude) than the heuristics [11].

Cahn argues that the best topology for a network is not limited to a tree. He discusses the strengths of a mesh network with multiple paths between the locations, rather than a tree, where the total cost may be lower. Mesh networks also result in decreasing delay and increasing link utilization [19:205]. Gurumohan [12] agrees that a

mesh topology is a good choice for network architecture. He finds that a mesh topology provides high availability, connectivity, increased capacity and network utilization. Several other researchers point to similar advantages of mesh topologies in their respective studies [10, 21, 22, 33, and 34].

Many topology design techniques have been developed for wired networks and wireless networks with omnidirectional antennas [12]. Acompora, Krishnamurthy, and Bloom [33] propose the use of recursive grids for formation of a mesh topology. Gurumohan [12] points out, however, that recursive grids do not take into account the random distribution of nodes and therefore is not a good method for finding topologies. Both Von Conta and Maekawa [35,36] pursue the formation of topologies with minimum diameter for networks used to interconnect several processors. Farago [37] proposed the use of random graphs to construct networks with strong connectivity and optimal diameter for a virtual private network topology. The preceding studies of designing network topologies are similar to the focus of the research presented in this paper, however directional hybrid wireless networks are considerably different and the results cannot be easily applied to them [12]. According to Gurumohan, the topology control and the design issues dealt with in multi-hop wireless networks are much more similar to the ones involved in free space-optical (FSO) networks, which are a type of directional wireless network [12].

Hu [10] studies the topology control of a multi-hop Packet Radio Network. Hu develops a topology by first constructing a backbone network with many edges using the Delaunay triangulation. The network is then optimized for achieving a uniform degree and high throughput by removing the edges that violate the degree constraint and the

communication range [10]. Although this method produces good topologies, it cannot be guaranteed that the network is connected in all cases. Gurumohan addresses this issue with his Closest Neighbor method [12:2], where he begins with a backbone network by constructing a dcMST. This guarantees connectivity in the network while satisfying the degree constraints for each node. Edges are then added to the tree, developing it into a mesh network. After the dcMST is constructed, nodes with degree below their upper bounds are picked in increasing order of their current degree. Edges are constructed with as many closest neighbors as possible while staying under the upper bound. In Gurumohan's study, closeness is based on the distance between nodes, but a general cost of constructing a link between nodes could be used instead of distance.

Mixed-Integer Linear Program Approach

The strength of the mixed-integer linear program (MILP) approach presented by Ahuja et al. [5] is that the solution to the MILP is optimal. This solution provides us with the minimum-cost topology and routing. The weakness, however, lies in the computational complexity of the problem formulation. The MILP does not scale well with a substantial increase in problem size.

MILP Solution Methods

There are several methods for solving linear program (LP) problems. Three of the more popular methods include the primal and dual simplex algorithms and the Newton Barrier interior point algorithm. The best algorithm to use, however, is problem-specific. Generally, the dual simplex algorithm is usually much faster than the primal simplex algorithm if the model is not infeasible or near infeasibility. The primal simplex method,

meanwhile, is usually the best choice for problems that are likely infeasible as it makes determining the cause of the infeasibility less difficult. Interior point methods such as the Newton Barrier algorithm perform better on certain classes of problems. This method, however, would likely perform slowly in situations where $A^T A$ is dense, where A is the LP constraint matrix [38:21].

The region defined by a set of linear constraints is known as the feasible region. The simplex method is based on the fact that the optimal solution to the LP lies on the boundary of the feasible region. Generally, simplex methods consider solutions at the vertices on the boundary of the feasible region and proceed from one vertex to another until an optimal solution has been found, or the problem proves to be infeasible or unbounded. The difference between the primal and dual simplex methods lies in which vertices they consider and how they iterate. The Newton Barrier method, however, is an interior point method. An interior point method involves iteratively moving from one point to the next within the interior of the feasible region. Approaching the boundary of the region is penalized, so the procedure cannot leave the region. Since the optimal solution of LP problems lie on the boundary of the feasible region, however, this penalty must progressively decrease as the algorithm continues in order to allow iterates to converge to the optimal solution. Since interior point methods usually give a solution lying strictly within the interior of the feasible region, this solution can only be an approximation to the true optimal vertex solution. Therefore, the desired nearness to the optimal solution, and not the number of decision variables, influences the number of iterations required to reach that solution. The Newton Barrier method often completes in

a similar number of iterations as the simplex method, regardless of problem size [38:21-23].

The Branch and Bound technique is often used to solve MILP problems. The first step in the process is a relaxation of the MILP where the integrality constraints are dropped. The relaxed problem is solved as a LP. If the LP is feasible, but the integrality constraints are not met, then more work is required. Unsatisfied integrality constraints are progressively selected and the concept of separation is applied, which can be depicted as a tree-searching algorithm. Each node of the tree is a MILP subproblem. At some point in the tree-searching procedure, an integer solution may be found, providing a bound on the solution to the MILP. If the value of the LP relaxation is not better than the cutoff, then branching from that node can be discontinued, for any descendant of the node cannot be better than the cutoff value already found. This concept of a cutoff value can be applied when no integer solution has been found if it is known, or it can be assumed from the beginning of the procedure that the optimal solution must be better than some value [38:25-26].

The Decomposition Principle

In order to mitigate some of the computational complexity issues inherent in the MILP formulations of the network design problem, one can employ relaxation or decomposition techniques. Bazaraa, Jarvis, and Sherali [6] instruct that, in regards to linear programming problems, Dantzig-Wolfe decomposition, Benders' partitioning, and Lagrangian relaxation are equivalent techniques. Decomposition is a methodical procedure for solving large-scale linear programs. It is especially useful for solving

problems with constraints having special structures such as found in many network flow problems. It is not necessary that the constraint set be of a special structure. If a special structure exists, however, we can take advantage of it for increased efficiency. The approach of the decomposition procedure is to operate on the two separate linear programs created by the two constraint sets. Information is passed back and forth between the two linear programs until the solution to the original problem is reached. The linear program from the general constraints is known as the master problem, and the linear program from the special constraints is called the subproblem.

Lagrangian Relaxation is one of a few solution methods in optimization that can be utilized in both linear and integer programming, combinatorial optimization, and nonlinear programming. Lagrangian Relaxation is a solution method that allows us to decompose a mathematical program and take advantage of its special structure. Therefore, this approach is very useful for solving many models with an embedded network structure [5]. Suppose that we consider the following general optimization model formulated in terms of a vector x of decision variables:

$$z^* = \min cx \tag{2.12}$$

subject to

$$Ax = b \tag{2.13}$$

$$x \in X \tag{2.14}$$

This model (P) has a linear objective function cx and a set $Ax = b$ of explicit linear constraints. The decision variables x are also constrained to lie in a given constraint set X which, for example, could model an embedded network flow structure. The

Lagrangian Relaxation procedure relaxes the explicit linear constraints by including them in the objective function with associated Lagrange multipliers λ. The resulting problem follows:

$$\text{Minimize } \ cx + \lambda(Ax - b) \tag{2.15}$$

subject to

$$x \in X \tag{2.16}$$

This new problem is called the Lagrangian Relaxation or Lagrangian Subproblem of the original problem. The function

$$L(\lambda) = \min\ \{cx + \lambda(Ax - b) :\ x \in X\} \tag{2.17}$$

is called the Lagrangian Function.

For any vector λ of the Lagrangian multipliers, the value $L(\lambda)$ for the Lagrangian function is a lower bound on the optimal objective function value z^* of the original optimization problem (P). To get the best possible lower bound on the original problem (P), we would need to solve the following problem:

$$L^* = \max_{\lambda} L(\lambda) \tag{2.18}$$

This is referred to as the Lagrangian multiplier problem associated with the original optimization problem (P) [5].

Lagrangian Relaxation is particularly powerful for the optimization of separable nonlinear programming problems or integer programming problems. The key idea of the approach is decomposition and coordination. The subgradient method is the most widely

used method, where the subgradient direction is obtained after all the sub-problems are solved and the multipliers are updated along this subgradient direction [8].

Network Metrics

A very important question that is inherent in the NDP and topology control problems is what makes one topology better than another one. What determines a good topology? Many different metrics can be found in the literature. The metrics used always incorporate design objectives, with additional metrics used to evaluate indirect consequences of a topology design strategy. An obvious network metric is its cost, including fixed construction cost and variable flow costs. Costs can be defined generally or specifically. While a certain method's computational time is necessarily compared to the time required by other methods, computational time is not a characteristic of the resulting network topology.

Cahn [19] contends that a good design is one with a relatively low average number of hops. This metric refers to the average number of links on the path between a source and destination node. If the average number of hops is too large, the traffic is likely taking a route from source to destination that is too indirect or roundabout. Many researchers use this metric to describe a network's connectivity or network delay [19,12,42,43,44,45,50]. Another metric that is often associated with a network's connectivity is its diameter. The diameter of a network topology is the maximum hop distance between all source-destination node pairs. The diameter of a topology provides a quantitative measure of a network's connectivity. A topology with small diameter is

considered better connected than one with a larger diameter [49]. Other researchers have used this metric as well [12,46,47,48,50].

Another important metric is the satisfaction of bandwidth requirements. A QoS connection, or commodity, request usually comes with a bandwidth requirement and the subsequent routing seeks a source to destination route with the requested bandwidth. If no such route can be found, the commodity request should be blocked. Therefore a network topology that satisfies a higher percentage of the commodity requests is a better one [40,41].

Summary

This chapter discussed the fundamental concepts and literature underlying the work presented in this thesis. We began with a review of communication networks and the network design problem. We discussed several instances of the network design problem and techniques that have been applied. We discussed in further detail the mixed-integer linear programming approach and different solution methods and decomposition techniques. We then reviewed common network topology metrics to compare the results of different topology design strategies. This chapter provides the foundation to develop the methodology discussed in Chapter III.

III. Methodology

The Problem

The Network Design Problem (NDP) considered in this research is unique. It differs from the research discussed in Chapter II with the addition of multiple types of interfaces at each node. In this case, a connection between two nodes can be made via a certain interface type only if both nodes contain compatible interface types. The number of connections that can be made at any given node is determined by the number of interfaces at that node. Each potential link has an associated fixed cost for including that link in the network topology. As presented in the NDP discussion by Ahuja, et al. [5:628], we also make the general assumption that a unique commodity exists for each possible source-destination pair of nodes in the network. Each commodity has an associated cost per unit flow over each possible link. We are given the number of nodes, number of types of interfaces, the number and types of interfaces available at each node, a list of all commodities with their respective node-destination pairs and bandwidth requirements, which links are possible, the fixed cost of including such links, the capacity of each link, and the cost per unit flow for each commodity on each link. Given these network characteristics and requirements, we seek a network topology that minimizes the total cost of the network (link cost and flow cost). The topology must satisfy traffic requirements and capacity constraints while ensuring degree and interface constraints are satisfied. We present a simple instance of this problem to illustrate its nature.

Before we present a simple instance of the NDP, we show a representative arc. We let u_{if} denote the number of interfaces of type f at node i. The fixed cost of including

an arc from node i to node j via interface type f in the network is denoted c_{ijf}. The capacity of each arc is given by cap_{ijf}. In the following arc representation we let the number of interface types be fixed at 2.

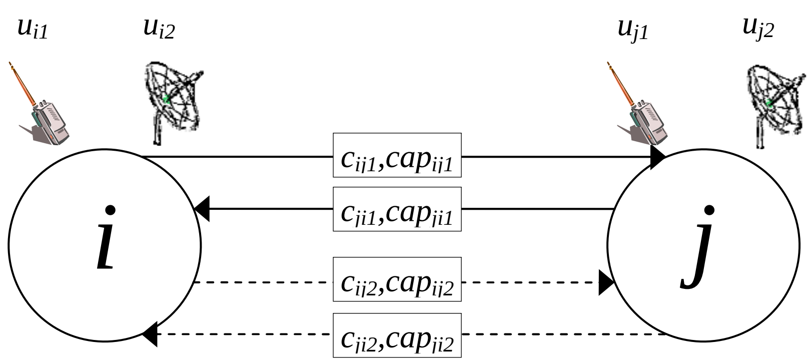

Figure 3.1. Representative Arc from the Network Design Problem

We use solid and dashed arcs to distinguish between the different interface types. We assume if node i is connected to node j by interface type f, then node j is connected to node i by the same interface type. The fixed cost and capacity associated with each link are not assumed to be equal. These assumptions are made to consider an intentionally broad and general case of this NDP. The variable cost per unit flow for each commodity over each arc is omitted, but not forgotten, from the representation for simplicity. Let us now consider a network with four nodes and two interface types, which we present in Figure 3.2.

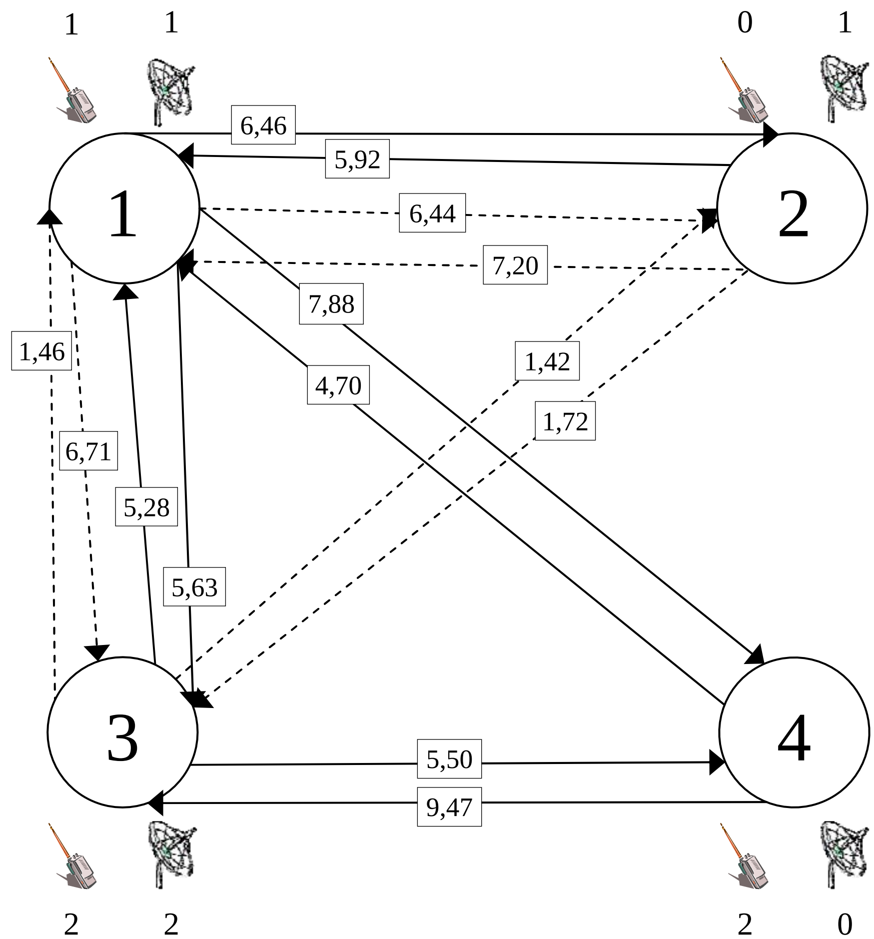

Figure 3.2. Four-Node Instance of the Network Design Problem

There are twelve commodities which correspond to the twelve possible source-destination pairs. In Table 3.1, we are given a list of these commodities along with each commodity's bandwidth requirement.

Table 3.1. Commodity List for Four-Node Instance of the NDP

Commodity	Source	Destination	Bandwidth Required
1	1	2	4
2	1	3	1
3	1	4	2
4	4	1	3
5	2	1	1
6	2	3	4
7	2	4	2
8	4	2	2
9	3	1	5
10	3	2	1
11	3	4	2
12	4	3	3

We do not assume that the capacity for traffic flowing from node i to node j is equal for traffic flowing from node j to node i. We therefore have different capacities for every directional arc, which are given in Figure 3.2.

Generating Network Characteristics

As the size of this problem increases, the amount of information needed for input grows dramatically. For example, a network with 5 nodes will have 20 commodities (one for each of the twenty possible source-destination node pairs), while a network with twice as many nodes (10) will have 90 commodities. In this 10-node network, assume there are three types of interfaces potentially at each node. Then the flow cost for each commodity over each possible link must be given. In a complete network, where each node can connect to every other node, there are $(10)\bullet(10) = 100$ possible links, so we would need $(90)\bullet(100)\bullet(3) = 27,000$ values just to describe the cost per unit flow for each commodity

along each potential link. Following this line of thought, we soon realize the necessity to develop a method to quickly generate data sets describing the characteristics of a network.

In order to maintain generality while testing the model, we produce the network characteristics randomly. We decide how many nodes and different interface types we want in the network along with a desired degree of incidence, expressed as a percentage. For example, suppose we desire a network with 10 nodes, 3 interface types, and 40% incidence. We then randomly generate how many of each type of interface is at each node. A list of the commodities with their source-destination pairs is created, and their respective bandwidth requirements are generated randomly. The node incidence matrix is randomly generated so that the network contains 40% of the possible number of links contained in a complete network. Due to the implied dynamic nature of this network, we generate link costs, capacities, and flow costs for each potential link even if that link is not currently available (according to the node-incidence matrix). We include the costs and capacities of links that are not available in case future events cause them to become available.

MILP Formulation

Ahuja, et al. [5] provide a MILP formulation of an uncapacitated NDP. We adapt this formulation by adding degree and interface constraints to accommodate the hybrid nature of this type of network. We also include capacity constraints to reflect the true physical nature of wireless telecommunications networks. Manipulating their formulation, we get the following model.

Let N denote the set of nodes, K the number of commodities, and F the number of interface types.

Let (i, j, f) denote the arc connecting node i to node j by interface type f.

Let A denote the node-incidence matrix where $a_{ijf} = 1$ if node i is incident to node j via interface type f, and $a_{ijf} = 0$ otherwise.

Let x_{ijf}^k denote the fraction of the required flow of commodity k to be routed from the source s^k to the destination d^k that flows on arc (i, j, f).

Let y_{ijf} denote the binary variable indicating whether arc (i, j, f) is selected as part of the network topology.

Let v_{ijf}^k denote the per unit cost for commodity k on arc (i, j, f) multiplied by the flow requirement for that commodity.

Let c_{ijf} denote the fixed cost of including arc (i, j, f) in the network.

Let u_{if} denote the number of interfaces of type f at node i.

Let b^k denote the the required bandwidth for commodity k.

Let cap_{ijf} denote the capacity of arc (i, j, f).

$$\text{Minimize} \quad \sum_{\{k,(i,j,f):\, a_{ijf}=1\}} v_{ijf}^k x_{ijf}^k + \sum_{\{(i,j,f):\, a_{ijf}=1\}} c_{ijf}\, y_{ijf} \tag{3.1}$$

subject to

$$\sum_{\{j,f:\, a_{ijf}=1\}} x_{ijf}^k - \sum_{\{j,f:\, a_{jif}=1\}} x_{jif}^k = \begin{cases} 1 & \text{if } i = s^k \\ -1 & \text{if } i = d^k \\ 0 & \text{otherwise} \end{cases} \qquad \forall\; i \in N, k = 1,\ldots,K \tag{3.2}$$

$$\sum_k r^k x_{ijf}^k \le cap_{ijf} \qquad \forall (i, j, f) \in A \;\ni a_{ijf} = 1 \tag{3.3}$$

$$\sum_{j\in N} y_{ijf} \le u_{if} \qquad \forall i \in N,\; f = 1,\ldots,F \tag{3.4}$$

$$x_{ijf}^{k} \le y_{ijf} \qquad \forall (i, j, f) \in A \ni a_{ijf} = 1, k = 1, \ldots, K \tag{3.5}$$

$$y_{ijf} = y_{jif} \qquad \forall (i, j, f) \in A \ni a_{ijf} = 1 \tag{3.6}$$

$$x_{ijf}^{k} \ge 0 \qquad \forall (i, j, f) \in A \ni a_{ijf} = 1,\ k = 1, \ldots, K \tag{3.7}$$

$$y_{ijf} \text{ is binary} \qquad \forall (i, j, f) \in A \ni a_{ijf} = 1 \tag{3.8}$$

Equation (3.2) implies $x_{ijf}^{k} \le 1$. Equation (3.3) represents the arc capacity constraints, and equation (3.4) represents the interface degree constraints. We assume that if node *i* is connected to node *j*, then network traffic can flow in each direction, that is, from *i* to *j* or from *j* to *i*, requiring equation (3.6).

A shortfall of this formulation is that it has a feasible solution only if a topology with sufficient link capacity to satisfy all commodity bandwidth requirements exists. Otherwise, there is no feasible topology, and the model yields no solution. In such a case, we add to the model the ability to selectively drop commodity constraints, to the extent a feasible solution can be found. In other words, we must exclude certain commodities to allow sufficient capacity to satisfy the bandwidth requirements of the remaining commodities. In this study, as in previous studies [40,41], we assume that there is no value in satisfying any less than 100% of a commodity's demand. Therefore an entire commodity is dropped rather than allowing partial satisfaction of the commodity's demand. An omitted commodity, though, equates to a failure to send information from one user to another, which is extremely undesirable. Therefore, if commodities must be dropped, we do so in increasing order of priority. In other words, we drop the lowest priority commodity first. We arbitrarily assume a commodity's

priority is directly proportional to its bandwidth requirement, that is higher bandwidth requirement equates to a higher priority level. We introduce an additional binary variable m^k, which denotes the decision to omit commodity k from consideration. If $m^k = 1$, then commodity k is dropped. A very large penalty is given in the objective function associated with dropping commodity k so that commodities will be dropped only to achieve feasibility. The revised formulation is as follows:

$$\text{Minimize} \quad \sum_{\{k,(i,j,f):\, a_{ijf}=1\}} v_{ijf}^k x_{ijf}^k + \sum_{\{(i,j,f):\, a_{ijf}=1\}} c_{ijf} y_{ijf} + \sum_k 1000 r^k m^k \tag{3.9}$$

Equation (3.2) is replaced by equation (3.10), and equation (3.11) is added. Equation (3.10) allows the omission of commodities.

$$\sum_{\{j,f:\, a_{ijf}=1\}} x_{ijf}^k - \sum_{\{j,f:\, a_{jif}=1\}} x_{jif}^k = \begin{cases} 1 - m^k & \text{if } i = s^k \\ -1 + m^k & \text{if } i = d^k \\ 0 & \text{otherwise} \end{cases} \quad \forall\ i \in N, k = 1, \ldots, K \tag{3.10}$$

$$m^k \text{ is binary} \qquad \forall k = 1, \ldots, K \tag{3.11}$$

Solving this MILP will provide an exact solution to the NDP, specifying which links should be included in the network and which links each commodity should flow on. The MILP can be solved using any linear solver application. We use XPRESS-MP, which can employ the dual simplex, primal simplex, or the Newton Barrier method to solve the relaxed LP. We use all three methods to determine if one out-performs the others with this type of problem.

Heuristic Approach

Significant changes in the network structure of a dynamic network may disable large portions of the network. Waiting thirty minutes, for example, to compute a new topology equates to even more time where many users of the network are virtually disconnected. We formulate a different solution strategy where we can find a good solution with much less computational effort. As shown in Chapter II, creating a mesh network with a backbone is a common approach to the NDP [10,11,12]. The dcMST is a good choice for the backbone, because it guarantees connectivity at a minimum cost [11,12]. After constructing a dcMST, we can then add edges to the network to form a mesh topology. Among edge-adding strategies, Premkumar and Chu [11] found that GA methods provide slightly better solution quality than commonly used heuristics, but other heuristics perform much faster. Starting with a dcMST, we use a heuristic algorithm to add edges and improve the network's QoS. Modifying the MST formulation presented by Guéret, et al. [51], we formulate the dcMST problem as an integer program (IP). The objective is to find a minimum cost network that guarantees connectivity.

Let $N = \{1, 2, ..., n_N\}$ be the set of all n_N nodes in the network.

Let $F = \{1, 2, ..., n_F\}$ denote the set of the n_F different types of interfaces used in the network.

Let A be the $(n_N \times n_N \times n_F)$ node-incidence matrix with $a_{ijf} = 1$ if node i is incident to node j by interface type f, and $a_{ijf} = 0$ otherwise.

Let y_{ijf} denote the binary decision variable indicating whether or not edge (i, j, f) is chosen. $y_{ijf} = 1$ if chosen. $y_{ijf} = 0$ otherwise.

Let c_{ijf} denote the cost of including edge (i, j, f) in the network.

Let u_{if} denote the number of interfaces of type f at node i.

Let $Level_i$ denote the integer value that corresponds to the number of links in the path from the root node to node i.

$$\text{Minimize } z = \sum_{\{(i,j,f):\, a_{ijf}=1\}} c_{ijf}\, y_{ijf} \tag{3.12}$$

subject to

$$\sum_{\{(i,j,f):\, a_{ijf}=1\}} y_{ijf} = (n_N - 1) \tag{3.13}$$

$$\sum_{j \in N} y_{ijf} + \sum_{j \in N} y_{jif} \le u_{if} \qquad \forall i \in N,\ f \in F \tag{3.14}$$

$$Level_j \ge Level_i + 1 - n_N + n_N \left(\sum_f y_{ijf}\right) \qquad \forall i, j \in N \ \ni \sum_f a_{ijf} \ne 0 \tag{3.15}$$

$$\sum_{\{(j,f):\, a_{ijf}=1\}} y_{ijf} = 1 \qquad \forall i \in \{2, 3, ..., n_N\} \tag{3.16}$$

$$y_{ijf} \text{ is binary} \qquad \forall (i, j, f) \in A \tag{3.17}$$

$$Level_i \ge 0 \text{ is integer} \qquad \forall i \in N \tag{3.18}$$

Equation (3.14) ensures satisfaction of the interface degree constraints. We use the XPRESS-MP optimizer to solve this IP formulation of the dcMST problem.

Once a backbone is found, links must be added to form a mesh network. Gurumohan [12] picks nodes with degree below their upper bounds in increasing order of their current degree. Links are then constructed with as many closest neighbors as possible while staying under the upper bound. We adopt a similar strategy for adding links to the dcMST. We first determine the gap between the current degree and the upper bound for each node. We then sequentially visit each node in non-decreasing order of that gap. Links are added to the network connecting each node to as many of its incident nodes as possible while satisfying the degree constraints of all nodes in consideration. With this strategy, we essentially add links by starting with the nodes with the most unused interfaces. We also consider an alternate strategy where the nodes are visited in non-increasing order of their respective degree gap. The link-adding heuristic strategy is outlined in Figure 3.3.

Find the dcMST

Determine the types and number of unused interfaces at each node, e_{if}.

Determine the total number of unused interfaces at each node, $ub_i = \sum_f e_{if}$

Sort the nodes in non-decreasing (non-increasing) order of number of unused interfaces.

Scan through the node list, adding as many links as possible while satisfying degree bounds.

Figure 3.3. Pseudo Code for Link-Adding Heuristic Strategy

Including a dcMST with the MILP Approach

The advantage of the MILP solution strategy is that it considers the network's traffic requirements when deciding the optimal topology. The heuristic approach, ignoring traffic requirements, only considers the fixed link cost in the dcMST construction phase, and adds links regardless of link or flow costs. An enormous disadvantage, however, for the MILP approach is its computational complexity. Relatively large amounts of time are required to find the solution to the MILP. Meanwhile, the main advantage of the heuristic approach is the short time required to generate the topology.

Our final solution strategy combines the dcMST and MILP approaches. A dcMST can be found quickly and included in the final network topology. The MILP formulation can then be used to determine which links to add to the dcMST. This approach saves time by beginning the MILP phase with a minimal solution that does not have a guarantee of optimality. We implement this approach by first finding a dcMST using the IP formulation discussed earlier. We then add the following constraint to the MILP formulation to include the dcMST in the final network topology.

$$y_{ijf} = 1 \qquad \forall (i, j, f) \in A \ni y_{ijf} \text{ is in the dcMST} \tag{3.19}$$

Summary

This chapter discussed the instance of the NDP that we consider in this research. We discussed the generation of experimental network requirements given a specified number of nodes, number of interfaces, and the degree of incidence. We discussed the three main solution strategies that will be investigated further in Chapter IV. We will

also implement the MILP approach with dual simplex, primal simplex, and Newton Barrier methods. We will also compare two different link-adding strategies in our heuristic approach. In our third approach we use the dcMST to provide the MILP approach with a partial solution in order to decrease its computational complexity.

IV. Results

Data Set Generation

To test our models introduced in Chapter III, we produce network characteristic data sets randomly. We test all of the solution strategies and use the results to highlight strengths and weaknesses for each strategy. The parameters for every data set are fixed throughout the testing. However, the number of nodes are varied to examine the effect of increasing the problem size. In cases where a random number is chosen from a certain interval, we assume the intervals are uniformly distributed. The number of interface types is set at three with the number of each type at a node picked randomly from the interval [0,3]. The node-incidence degree is set to 25%, that is, 25% of the maximum number of possible links are available to the network. A list of the commodities with their source-destination pairs is created, and their respective bandwidth requirements are generated randomly from the interval [1,5]. Due to the implied dynamic nature of this network, we generate link costs, capacities, and flow costs for each potential link even if that link is not currently available (according to the node-incidence matrix). Both the fixed link costs and the flow costs are picked from the interval [1,9], and the link capacities are chosen from the interval [10,99]. We include the costs and capacities of links that are not available in case future events cause them to become available.

In the process of producing these data sets, the explosion in problem size as the number of nodes increases becomes apparent in the size of the files generated. Figure 4.1 illustrates the corresponding increase in file size with an increase in the number of nodes.

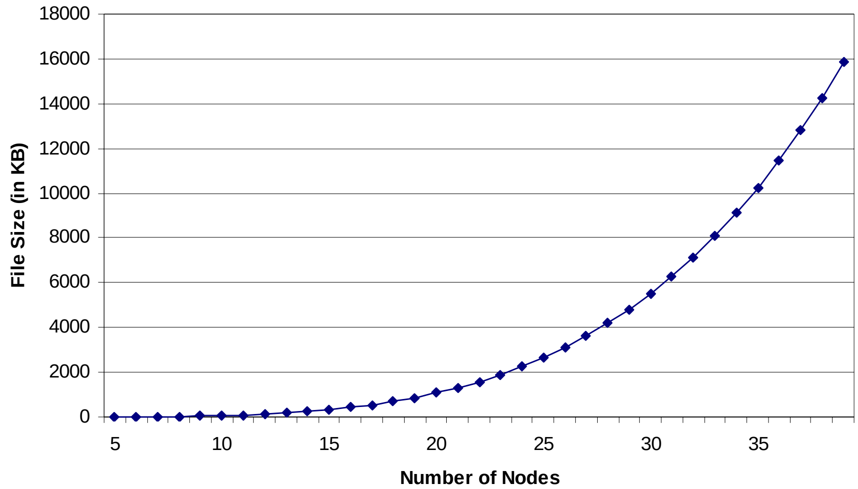

Figure 4.1. File Sizes for Numbers of Nodes (5 to 39)

From Figure 4.1, we see a dramatic increase in file size for any problem with more than 25 nodes. The rate of increase in file size appears to be exponential in nature, with respect to the number of nodes. Every attempt to generate an instance with 40 or more nodes fails due to insufficient memory. We therefore generate data sets for instances with 5 nodes up to 39 nodes.

Testing

We test the proposed strategies for problems with 5, 10, 15, 20, 25, 30, and 39 nodes. This allows us to cover the range allowed by data generation limits. Due to possible variance in performance measures, more than a few tests of each problem size must be conducted. Therefore, for each problem size we generate ten data sets for testing. Our testing is performed on an IBM Think Pad with 2.8 GHz, 512 MB DDR

RAM, 8 kB Level 1 Cache, 512 kB Level 2 Cache, and a 30 GB hard drive. The detailed results of all tests are shown in Appendix A.

Another Commodity Prioritization Approach

In Chapter III, we described the prioritization method for determining which commodities to consider for omission first. We drop the lowest priority commodity first, assuming a commodity's priority is directly proportional to its bandwidth requirement. Another possible way to prioritize the commodities is to give a preferential ordering, where the commodities are listed in non-increasing order of priority. This list provides a pre-emptive ordering of the commodities. Therefore, if a commodity must be dropped due to insufficient link capacity, all lower priority commodities must be dropped first. The objective function for the MILP formulation of the problem then becomes:

$$\text{Minimize} \quad \sum_{\{k,(i,j,f):\, a_{ijf}=1\}} v_{ijf}^k x_{ijf}^k + \sum_{\{(i,j,f):\, a_{ijf}=1\}} c_{ijf} y_{ijf} + \sum_k 1000 m^k \tag{4.1}$$

We then add the following constraints:

$$m^k \le m^{k+1} \qquad \forall k = 1,\dots,K \tag{4.2}$$

Equation (4.2) ensures a commodity is not dropped unless all lower priority commodities have been dropped already.

The main disadvantage with this prioritization method lies in its pre-emptive nature, which leads to undesirable results. The cases where this weakness is most apparent are ones where there are bottleneck links. Such links have insufficient capacity to accommodate a relatively high priority commodity, but do not service some subset of low priority commodities. That priority commodity must be dropped to provide enough

capacity to satisfy the bandwidth requirements, however, Equation (4.2) requires all commodities with a lower priority to be dropped as well, yielding a highly inefficient problem solution. In such cases, far too many commodities are needlessly dropped to consider the solution acceptable. In using this pre-emptive prioritization method, several test runs resulted in solutions where 40-80% of the commodities were dropped. We therefore discarded this approach from further testing, and prioritized the commodities according to their bandwidth requirements, as given in Chapter III.

Heuristic Post-Processing

The purpose of this research is to develop and evaluate several methods for providing a good topology for this special instance of the NDP. Of the various methods presented in Chapter III, the two heuristic solution strategies do not consider the commodity traffic requirements while constructing a mesh network topology. Therefore, to compare the various strategies, we can only examine the topology cost and the time required to produce it. In order to compare other QoS metrics such as network diameter, average number of hops, and number of dropped commodities, we institute a post-processing MILP for both heuristics. Because the heuristic gives a network topology, only the commodity flows must subsequently be determined. We use the same formulation as the MILP solution strategy, but omit the fixed link costs in the objective function. The only necessary constraints are the node balance constraints of Equation (3.10), the link capacity constraints of Equation (3.3), the non-negativity constraints of Equation (3.7), and the binary constraints of Equation (3.11), giving the following formulation for the heuristic post-processing MILP.

$$\text{Minimize} \quad \sum_{\{k,(i,j,f):\, a_{ijf}=1\}} v_{ijf}^{k} x_{ijf}^{k} + \sum_{k} 1000 r^{k} m^{k} \tag{4.3}$$

subject to

$$\sum_{\{j,f:\, a_{ijf}=1\}} x_{ijf}^{k} - \sum_{\{j,f:\, a_{jif}=1\}} x_{jif}^{k} = \begin{cases} 1-m^{k} & \text{if } i = s^{k} \\ -1+m^{k} & \text{if } i = d^{k} \\ 0 & \text{otherwise} \end{cases} \quad \forall\ i \in N, k = 1,\ldots,K \tag{4.4}$$

$$\sum_{k} r^{k} x_{ijf}^{k} \le cap_{ijf} \qquad \forall (i, j, f) \in A \ \ni a_{ijf} = 1 \tag{4.5}$$

$$x_{ijf}^{k} \ge 0 \qquad \forall (i, j, f) \in A \ \ni a_{ijf} = 1,\ k = 1,\ldots,K \tag{4.6}$$

$$m^{k} \text{ is binary} \qquad \forall k = 1,\ldots,K \tag{4.7}$$

Using this post-processing formulation allows a fair comparison of all metrics discussed in Chapter II.

Computational Complexity

The computational complexity of a solution strategy is important as it drives the amount of time required to give a solution to the problem. It is especially significant in dynamic time-sensitive environments such as a telecommunications network in a military combat zone. In such circumstances, the time required to develop a solution to the NDP becomes key to the continuity, stability, and eventual success of the mission. We must, therefore, compare the computational time for each solution strategy.

The two link-adding heuristic strategies and the dcMST/MILP combination method are expected to outperform the pure MILP formulations in computational time, but provide inferior topologies in terms of the number of dropped commodities and topology cost. The only difference between the three MILP solution strategies is the

method used to solve the relaxed LP. We compare these strategies for the sole purpose of determining whether one method outperforms the others.

As the number of nodes in the NDP increases, the complexity of the problem increases dramatically, as previously shown in this chapter. The computational time also increases dramatically. The optimal solution to the 15-node instance is not found within 8 hours of runtime. This is obviously unacceptable. Thus for problems with 15 nodes, we must stop the MILP search before it reaches an optimal solution. We do so by imposing a limit on the gap between the best integer solution and the best known lower bound found in the Branch-and-Bound process. For example, the solution search can be set to terminate when the best integer solution found is within, say, 5% of the best known bound. We assume the limit on required computational time must be no greater than 30 minutes to meet the communications demands in a dynamic environment. These scaling issues are expected, as Premkumar and Chu [36] limited their problem size to 12 nodes.

Premkumar and Chu encounter similar scaling issues in their work and limit their maximum problem size to 12 nodes, above which they can not find an optimal solution. In our testing, a solution for all ten test runs is found within the time limit when we impose a gap limit of 7% for Newton Barrier and Dual Simplex methods and a limit of 12% for the Primal Simplex method. In the test runs for the 20-node instance of the problem, each MILP formulation fails to find a feasible integer solution within the time limit, and the dcMST/MILP combination method requires a gap limit of 10% to find a solution for each test run. Since we cannot test the MILP formulations for instances with 20 nodes or more, we test only the remaining three strategies. The dcMST/MILP combination strategy suffers the same fate for problems with 25 nodes or more, so for the

25-node problem set we can only test the two heuristic strategies. At 35 nodes, the post-processing MILP fails to find a feasible integer solution within the time limit, so we continue testing without post-processing. Because this causes the absence of a QoS metric, we compare the number of connections made in each strategy's topology.

First, we compare the average computational time required by each of the different MILP methods.

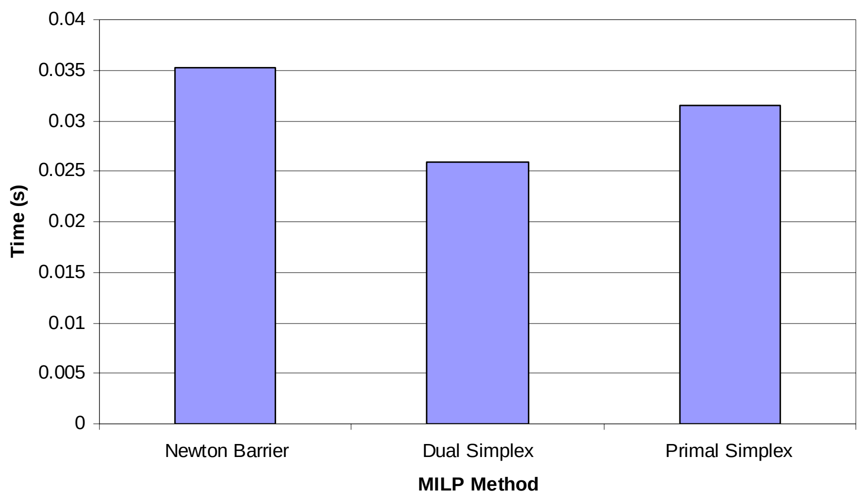

Figure 4.2. Time Comparison for MILP Methods (5 Nodes)

Every test run for the 5-node data set terminated within 1 second. Therefore, any difference in computational time here is trivial, and we proceed to the 10-node instance.

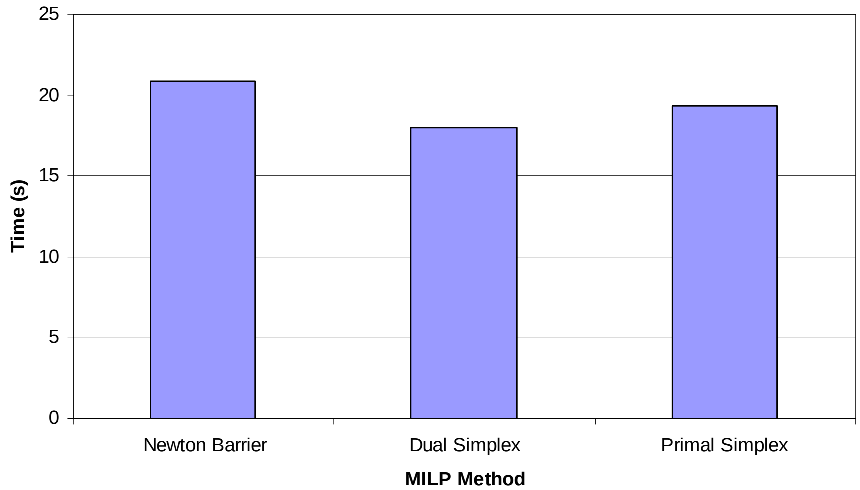

Figure 4.3. Time Comparison for MILP Methods (10 Nodes)

In Figure 4.4, the Dual Simplex method appears to dominate the other two methods. We perform a paired t-test (with $\alpha = 0.05$) to determine if there is a statistical difference between the mean computational time for the Dual Simplex method and the Primal Simplex method, which appears to the second best option.

Table 4.1. Paired t-Test for Mean Time with Dual Simplex and Primal Simplex Methods (10 Nodes)

	Dual	*Primal*
Mean	17.967	19.312
Variance	148.728	150.807
Observations	10	10
Hypothesized Mean Difference	0	
df	9	
t Stat	-1.070	
P(T<=t) one-tail	0.156	
t Critical one-tail	1.833	
P(T<=t) two-tail	0.312	
t Critical two-tail	2.262	

The p-values of both the one- and two-tail tests are well above alpha value of 0.05, therefore we cannot reject the null hypothesis that there is no difference in the means of the two sets. In other words, we cannot conclusively determine which method requires less time. Table 4.3 shows that there is no statistically significant difference between the means of the Dual Simplex and Newton Barrier methods.

Table 4.2. Paired t-Test for Mean Time with Dual Simplex and Newton Barrier Methods (10 Nodes)

	Dual	*Barrier*
Mean	17.967	20.893
Variance	148.728	368.907
Observations	10	10
Hypothesized Mean Difference	0	
df	9	
t Stat	-1.049	
P(T<=t) one-tail	0.161	
t Critical one-tail	1.833	
P(T<=t) two-tail	0.322	
t Critical two-tail	2.262	

The variance in the amount of computational time for each method is too great to draw any inferences about differences in performance.

For the 15-node instance of the NDP, recall that we imposed optimality gap limits to ensure each test run will terminate within 30 minutes. The Newton Barrier and Dual Simplex methods are within 7% of the optimal and the Primal Simplex method is within 12% of the optimal. The times presented in Figure 4.5 are the times required to find these suboptimal solutions.

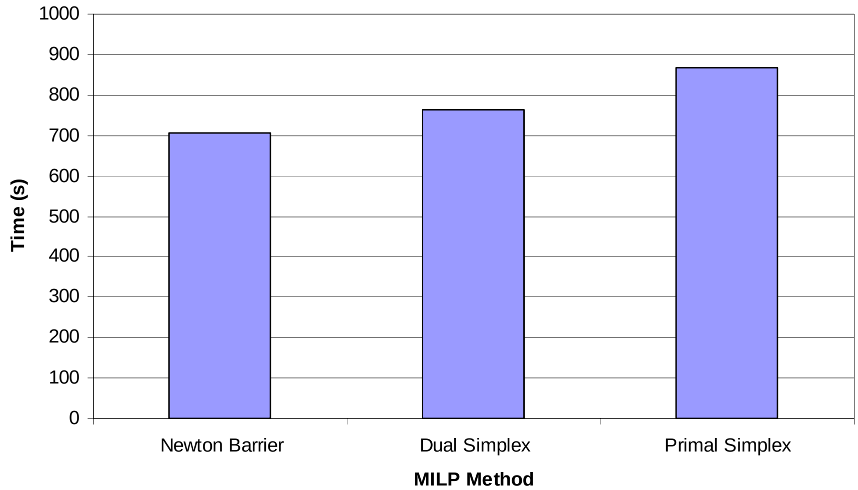

Figure 4.4. Time Comparison for MILP Methods (15 Nodes)

Even with the relaxed optimality gap of 12%, the Primal Simplex method seems inferior to the other two, that is, the Primal Simplex method requires more time to produce a worse solution. Table 4.4 shows the results of the t-tests comparing the Primal Simplex method to the other two.

Table 4.3. Paired t-Tests for Mean Time Comparing Primal Simplex with Dual Simplex and Newton Barrier Methods (15 Nodes)

	Primal	*Dual*		*Barrier*
Mean	867.871	762.572		707.27
Variance	568208.025	536890.901		481626.1
Observations	10	10		10
Hypothesized Mean Difference	0			0
df	9			9
t Stat	1.817			2.392
P(T<=t) one-tail	0.051			0.020
t Critical one-tail	1.833			1.833
P(T<=t) two-tail	0.103			0.040
t Critical two-tail	2.262			2.262

With $\alpha = 0.06$, α is greater than the p-value for the one-tail test with the Dual Simplex method, meaning we reject the hypothesis of equal means. With 94% confidence, we can say that the Dual Simplex method outperforms the Primal Simplex method. The p-values for both the one- and two-tail tests with the Newton Barrier method are lesser than $\alpha = 0.05$. Thus, we can say the Newton Barrier method outperforms the Primal Simplex method. The Newton Barrier method also appears to be superior to the Dual Simplex method.

Table 4.4. Paired t-Test for Mean Time with Newton Barrier and Dual Simplex Methods (15 Nodes)

	Barrier	*Dual*
Mean	707.270	762.572
Variance	481626.090	536890.901
Observations	10	10
Hypothesized Mean Difference	0	
df	9	
t Stat	-1.564	
P(T<=t) one-tail	0.076	
t Critical one-tail	1.833	
P(T<=t) two-tail	0.152	
t Critical two-tail	2.262	

If we let $\alpha = 0.08$ for the one-tail test, we conclude the Newton Barrier method also outperforms the Dual Simplex method.

As expected, the MILP formulations do not scale well with an increase in problem size. Figure 4.6 shows that the time required to obtain a solution explodes between 10 and 15 nodes. This explosion explains why a feasible integer solution could not be found within 30 minutes for problems with more than 15 nodes.

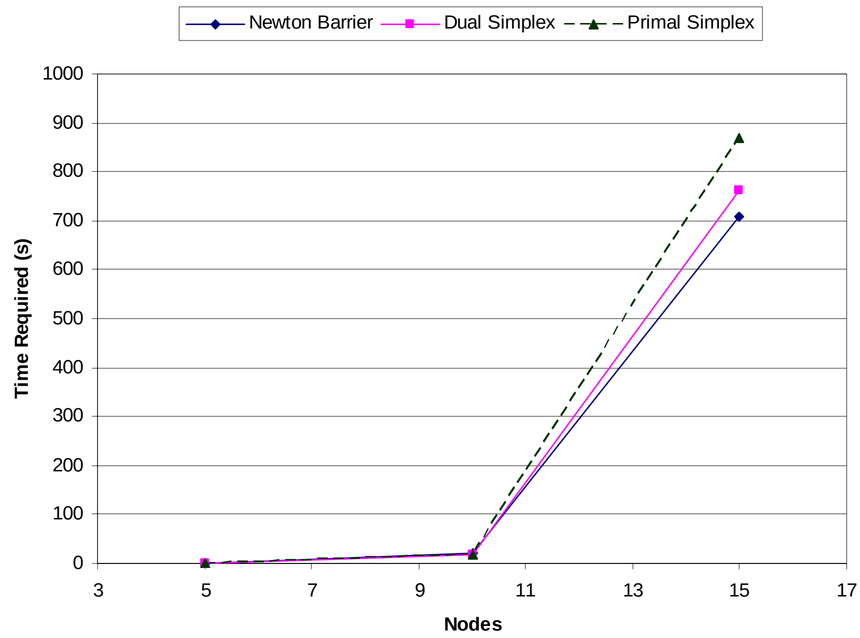

Figure 4.5. Number of Nodes vs. Time Required for MILP Formulations (5-15 Nodes)

Figure 4.7 gives a comparison of how the dcMST/MILP combination method scales compared to the pure MILP methods.

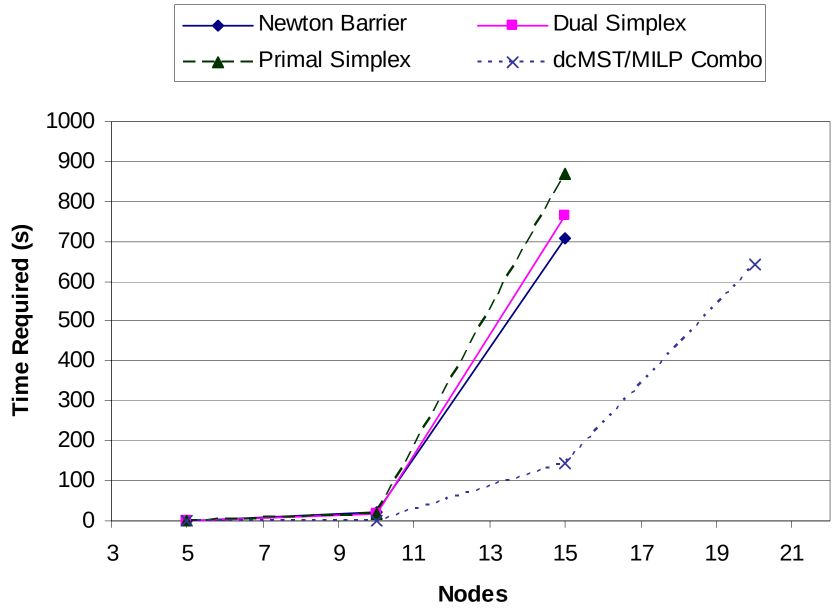

Figure 4.6. Number of Nodes vs. Time Required for MILP Formulations and Combination Method (5-20 Nodes)

Like the MILP formulations, the required time for the dcMST/MILP combination method increases dramatically as a function of the number of nodes. However, the severe increase occurs past 15-node size problems. From the p-values in Table 4.6, we see that the dcMST/MILP combination method outperforms all of the MILP formulations with $\alpha = 0.05$.

Table 4.5. Paired t-Tests for Mean Time Between the Combination Method and the MILP Methods (15 Nodes)

	Combo	*Barrier*	*Dual*	*Primal*
Mean	142.94	707.27	762.57	867.87
Variance	30546.61	481626.09	536890.90	568208.03
Observations	10	10	10	10
Hypothesized Mean Difference	0		0	0
df	9		9	9
t Stat	-2.394		-2.511	-2.823
P(T<=t) one-tail	0.020		0.017	0.010
t Critical one-tail	1.833		1.833	1.833
P(T<=t) two-tail	0.040		0.033	0.020
t Critical two-tail	2.262		2.262	2.262

For instances with 5-30 nodes, the time required by the heuristic methods to produce a topology never exceeds 3 seconds. The total time attributed to the heuristic methods consists almost entirely of the time required by the post-processing MILP formulation. The heuristic methods are far superior in computational time to the other methods. Figure 4.5 compares how the heuristic methods scale versus the dcMST/MILP combination method. Recall that Heuristic 1 adds links by visiting the nodes in non-decreasing order of the gap between the current degree and the degree upper bound. Heuristic 2 adds links in a similar manner, but visits the nodes in non-increasing order of the gap between the current degree and the degree upper bound.

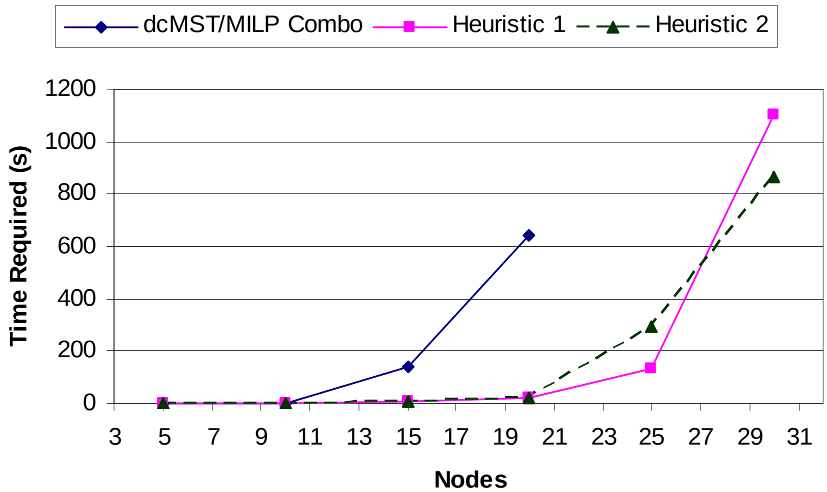

Figure 4.7. Number of Nodes vs. Time Required for dcMST/MILP Combination Method and Heuristic Methods (5-30 Nodes)

As expected, the scaling behavior exhibited by the heuristic strategies mimics that of the combination method, but the dramatic increase in time occurs at larger problem sizes. No distinguishable difference between the scaling behaviors of the two heuristic strategies is apparent.

We now compare the time each heuristic requires to construct its topology (ignoring the time required for the MILP post-processing).

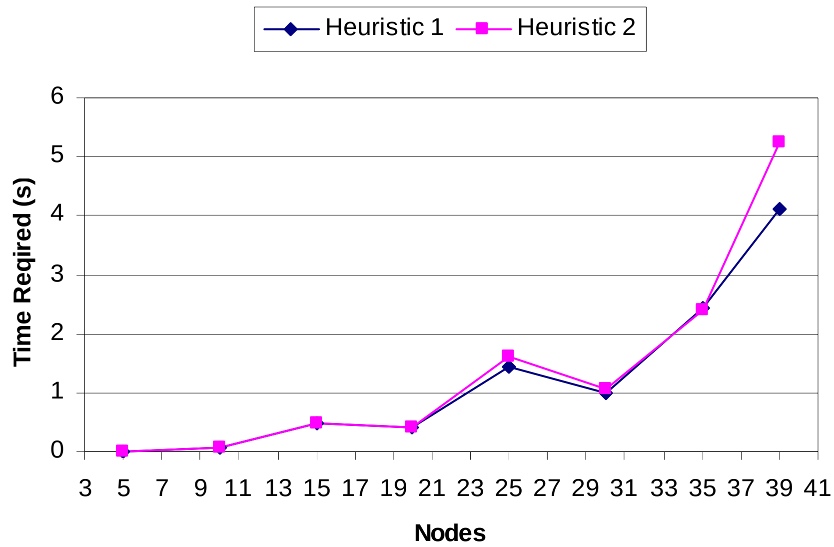

Figure 4.8. Number of Nodes vs. Topology Time Required for the Heuristic Methods (5-39 Nodes)

Up to the 35-node instance, Figure 4.9 shows little difference in mean topology times. The most separation between the mean times occurs after 35 nodes. We perform a statistical comparison of the computational time for the 39-node instance. The results of the paired t-test is shown in Tables 4.7.

Table 4.6. Paired t-Test for Mean Topology Time Between Heuristic 1 and Heuristic 2 (39 Nodes)

	Heuristic 1	*Heuristic 2*
Mean	4.114	5.261
Variance	3.440	1.244
Observations	10	10
Hypothesized Mean Difference	0	
df	9	
t Stat	-1.788	
P(T<=t) one-tail	0.054	
t Critical one-tail	1.833	
P(T<=t) two-tail	0.107	
t Critical two-tail	2.262	

If we let $\alpha = 0.06$, we can reject the null hypothesis for the one-tailed test. There is a statistical difference between the means, with Heuristic 1 outperforming Heuristic 2 on average.

Dropped Commodities

The main objective of the NDP is to minimize the total topology cost, including the fixed link costs and the variable flow costs. The other objective is to satisfy all of the users' demands. As stated in Chapter III, capacitated networks often cannot satisfy all of the traffic requirements, or commodity demands. Therefore a penalty for dropping commodities was added to the objective function in Equation (3.9), ensuring the least number of commodities are dropped to achieve feasibility. This penalty, however, is imposed for the sole purpose of minimizing the number of dropped commodities, but does not affect the actual cost to build and use the network. The MILP formulation is expected to outperform the other strategies in this metric. We compare the results of each strategy to determine the extent of this performance gap.

In the 5- and 10-node instances, the MILP methods produce the same solution for every data set, and no commodities are dropped. In the 15-node instance, as previously stated, the solutions found are suboptimal. Both the Newton Barrier and Dual Simplex methods terminate when a solution is found within 7% of the best known bound. The p-values found in Table 4.8 show no significant difference in the mean number of dropped commodities for these two methods.

Table 4.7. Paired t-Tests for Mean Number of Dropped Commodities Between the Newton Barrier and Dual Simplex Methods (15 Nodes)

	Barrier	*Dual*
Mean	5.8	5.7
Variance	43.956	42.011
Observations	10	10
Hypothesized Mean Difference	0	
df	9	
t Stat	1	
P(T<=t) one-tail	0.172	
t Critical one-tail	1.833	
P(T<=t) two-tail	0.343	
t Critical two-tail	2.262	

The Primal Simplex method terminates when a solution is found within 12% of the best known bound, but drops the same number of commodities as the Newton Barrier Method, on average.

Table 4.8. Paired t-Tests for Mean Number of Dropped Commodities Between the Primal Simplex Method and the Newton Barrier and Dual Simplex Methods (15 Nodes)

	Primal	*Barrier*		*Dual*
Mean	5.8	5.8		5.7
Variance	45.733	43.956		42.011
Observations	10	10		10
Hypothesized Mean Difference	0			0
df	9			9
t Stat	0.000			0.429
P(T<=t) one-tail	0.500			0.339
t Critical one-tail	1.833			1.833
P(T<=t) two-tail	1.000			0.678
t Critical two-tail	2.262			2.262

With the additional p-values in Table 4.9, we find no significant difference between the mean number of dropped commodities among all three MILP methods. Since these

methods have comparable variances, we use the mean from the Newton Barrier method to compare the other solution strategies with the MILP methods.

In the 5-node instance of the problem, none of the solution methods yield any dropped commodities. In the 10-node instance, 2 test runs (out of the 10) yield an omission of commodities for the combination method and Heuristic 1. Heuristic 2 yields dropped commodities in 5 of the test runs. We compare the means with the results from the MILP methods in Table 4.10.

Table 4.9. Paired t-Tests for Mean Number of Dropped Commodities Between the MILP Methods and the dcMST/MILP Combination and Heuristic Methods (10 Nodes)

	MILP	*Combo*		*Heuristic 1*		*Heuristic 2*
Mean	0	1.3		1.2		3
Variance	0.000	10.233		8.400		14.222
Observations	10	10		10		10
Hypothesized Mean Difference	0			0		0
df	9			9		9
t Stat	-1.285			-1.309		-2.516
P(T<=t) one-tail	0.115			0.111		0.017
t Critical one-tail	1.833			1.833		1.833
P(T<=t) two-tail	0.231			0.223		0.033
t Critical two-tail	2.262			2.262		2.262

Statistically, there is no discernable difference in means between the MILP methods and the Combination and Heuristic 1 methods. The MILP does, however, outperform Heuristic 2.

Table 4.11 compares the means of the MILP methods with those of the other methods for the 15-node instance of the problem.

Table 4.10. Paired t-Tests for Mean Number of Dropped Commodities Between the MILP Methods and the dcMST/MILP Combination and Heuristic Methods (15 Nodes)

	MILP	*Combo*	*Heuristic 1*	*Heuristic 2*
Mean	5.8	8.6	16.8	19
Variance	43.956	156.933	363.067	457.556
Observations	10	10	10	10
Hypothesized Mean Difference	0		0	0
df	9		9	9
t Stat	-1.290628676		-2.42097935	-2.5500139
P(T<=t) one-tail	0.114500109		0.019274334	0.015597017
t Critical one-tail	1.833112923		1.833112923	1.833112923
P(T<=t) two-tail	0.229000219		0.038548667	0.031194034
t Critical two-tail	2.262157158		2.262157158	2.262157158

The MILP methods statistically outperform both heuristics, but we fail to reject the hypothesis of equal means between the MILP methods and the dcMST/MILP combination method.

For the 20-node instance of the problem, we compare the number of dropped commodities between the dcMST/MILP combination method and the heuristic methods. As previously stated, the combination method here terminates upon finding a solution within 10% of the best known bound.

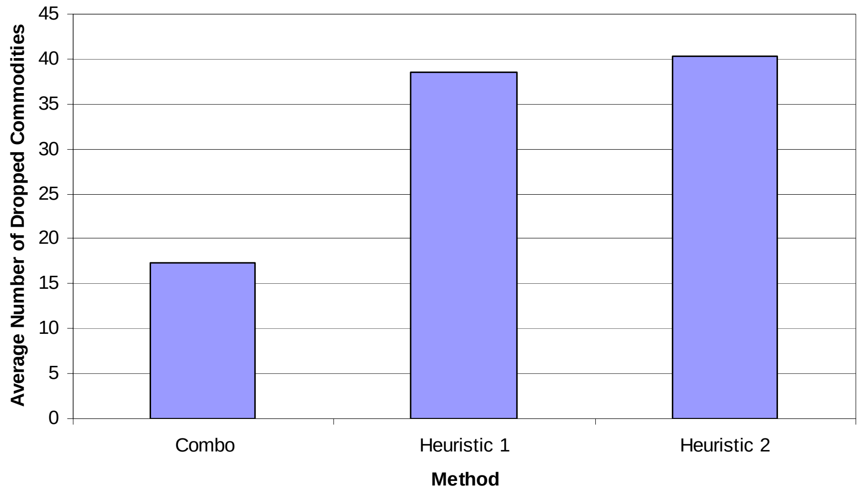

Figure 4.9. Average Number of Dropped Commodities Comparison for the dcMST/MILP Combination Method and the Heuristic Methods (20 Nodes)

We confirm that the suboptimal combination method solution dominates the heuristic solutions in Table 4.12.

Table 4.11. Paired t-Tests for Mean Number of Dropped Commodities Between the dcMST/MILP Combination Method and the Heuristic Methods (20 Nodes)

	Combo	*Heuristic 1*		*Heuristic 2*
Mean	17.4	38.5		40.3
Variance	385.822	737.833		614.233
Observations	10	10		10
Hypothesized Mean Difference	0			0
df	9			9
t Stat	-4.143			-4.185
P(T<=t) one-tail	0.001			0.001
t Critical one-tail	1.833			1.833
P(T<=t) two-tail	0.003			0.002
t Critical two-tail	2.262			2.262

In Figure 4.11, we compare the average number of dropped commodities by the two heuristic methods for the 25- and 30-node instances.

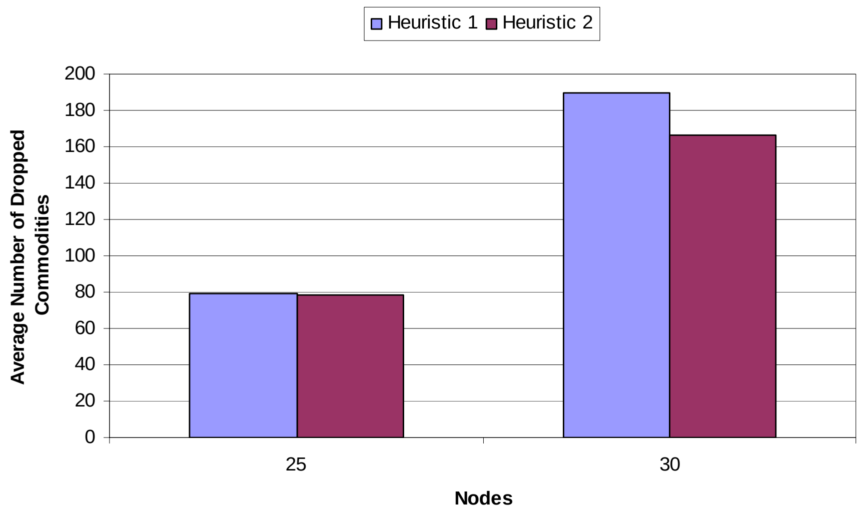

Figure 4.10. Average Number of Dropped Commodities Comparison for the Two Heuristic Methods (25 and 30 Nodes)

The difference between the means for the 25-node instance is very small. We examine the difference between the means for the 30-node instance in Table 4.13.

Table 4.12. Paired t-Test for Mean Number of Dropped Commodities Between Heuristic 1 and Heuristic 2 (30 Nodes)

	Heuristic 1	*Heuristic 2*
Mean	197.5	181.3
Variance	5920.94	3286.23
Observations	10	10
Hypothesized Mean Difference	0	
df	9	
t Stat	0.784	
P(T<=t) one-tail	0.227	
t Critical one-tail	1.833	
P(T<=t) two-tail	0.453	
t Critical two-tail	2.262	

According to the results of the paired t-test, we cannot deduce a significant difference between the means produced by the heuristic strategies.

In addition to the pair-wise comparisons, we show the scaling behavior for each method in Figure 4.12. In this case, we represent the number of dropped commodities as a percentage of the total number of commodities for each instance.

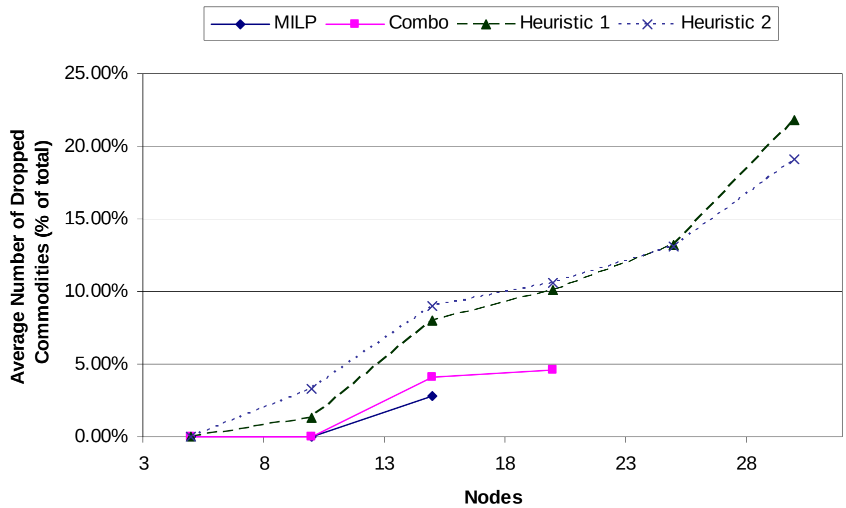

Figure 4.11. Average Number of Dropped Commodities as a Percentage of Total Number of Commodities for All Methods (5-30 Nodes)

Solution Quality (Topology Cost)

Solution quality is an obvious metric to use for strategy comparison. The total cost of a topology is comprised of the fixed link cost and the variable flow cost. The penalty in the objective function associated with dropping commodities is not representative of an actual network cost, but rather a means to minimize the number of commodities dropped. Therefore, in comparing the costs of the topologies produced by the different solution strategies, we compare only the costs associated with the construction and usage of the network topology excluding the penalties amassed by the omission of commodities. In cases where each strategy satisfies all (or almost all) of the commodity demands, the total topology costs can be fairly compared. If, however, a certain strategy provides a topology where, say, 10% of the commodities are dropped, the actual cost of this topology may be lower than the cost of one where most commodity

demands are met. Keeping this in mind, we compare the total topology costs provided by each of the solution strategies.

The only difference of total cost between the MILP methods occurs in the 15-node instance of the problem, where we terminate each method before reaching an optimal solution. Each method reaches the same optimal solution for the 5- and 10-node instances. The differences in cost for the 15-node instance are shown in Figure 4.13.

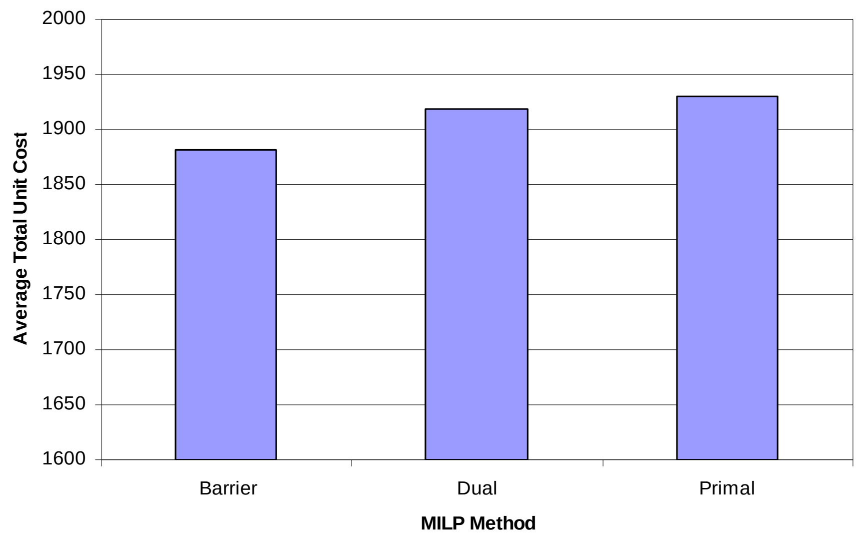

Figure 4.12. Comparison of Mean Total Unit Cost for the MILP Methods (15 Nodes)

We compare the mean total cost provided by the Netwon Barrier method with the other two methods in Table 4.14.

Table 4.13. Paired t-Test for Mean Total Unit Cost Between the Newton Barrier method and the Dual and Primal Simplex Methods (15 Nodes)

	Barrier	*Dual*	*Primal*
Mean	1881.21	1918.31	1930.40
Variance	11854.49	21064.94	24732.74
Observations	10	10	10
Hypothesized Mean Difference	0		0
df	9		9
t Stat	-1.670		-1.662
P(T<=t) one-tail	0.065		0.065
t Critical one-tail	1.833		1.833
P(T<=t) two-tail	0.129		0.131
t Critical two-tail	2.262		2.262

We can say on average, the Newton Barrier method provides solutions superior to the Dual and Primal Simplex methods with $\alpha = 0.07$.

The average number of dropped commodities for the 5- and 10-node instances for the dcMST/MILP combination method is 0 and 1.3, respectively. Thus cost from the combination method can be fairly compared to the costs from the MILP formulations which do not omit any commodities for both instances. Similarly, the two heuristic strategies can also be compared with the others for the 5- and 10-node instances.

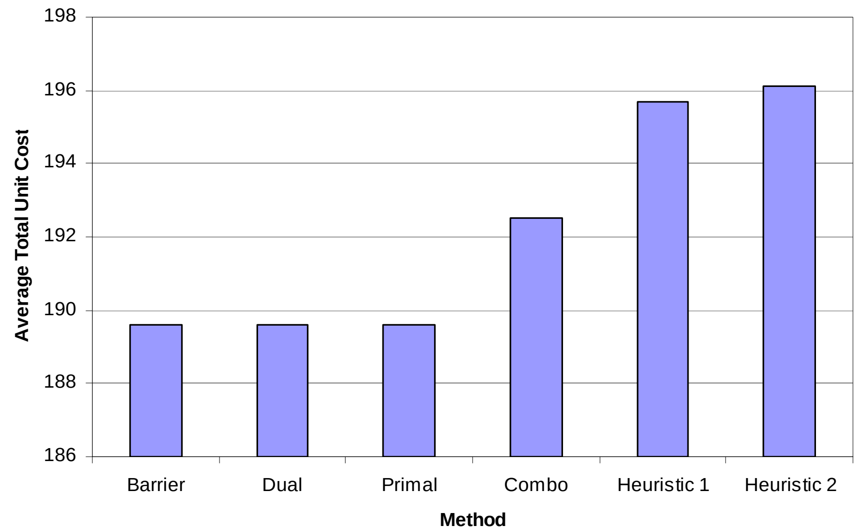

Figure 4.13. Comparison of Mean Total Unit Cost for All Methods (5 Nodes)

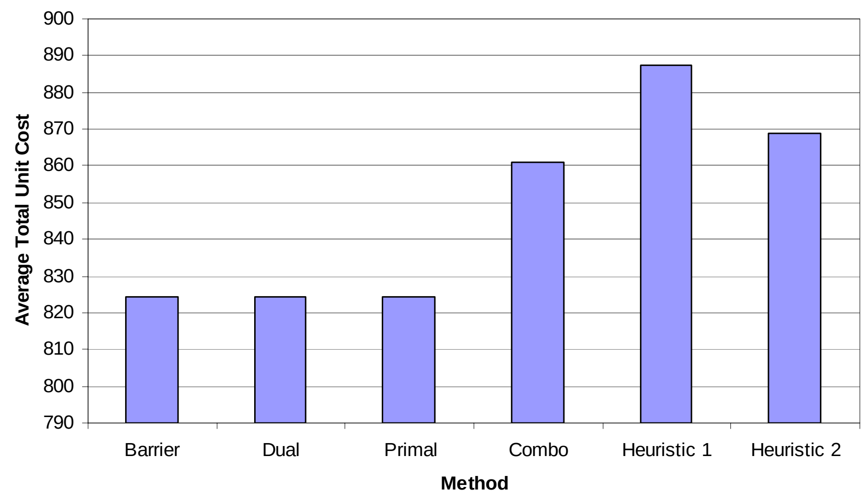

Figure 4.14. Comparison of Mean Total Unit Cost for All Methods (10 Nodes)

In the both the 5- and 10-node instances, each MILP method gives the same solution. We compare the mean cost of the remaining three methods to see if we can determine solution dominance.

Table 4.14. Paired t-Test for Mean Total Unit Cost from the MILP Method with the Means from the dcMST/MILP Combination and Heuristic Methods (5 Nodes)

	MILP	*Combo*		*Heuristic 1*		*Heuristic 2*
Mean	189.6	192.5		195.7		196.1
Variance	302.49	264.94		300.68		318.54
Observations	10	10		10		10
Hypothesized Mean Difference	0			0		0
df	9			9		9
t Stat	-2.301			-7.672		-6.960
P(T<=t) one-tail	0.023			0.000		0.000
t Critical one-tail	1.833			1.833		1.833
P(T<=t) two-tail	0.047			0.000		0.000
t Critical two-tail	2.262			2.262		2.262

All of the p-values in Table 4.14 are less than $\alpha = 0.05$, indicating the solution to the MILP formulation dominates the solutions of the combination and heuristic strategies.

Table 4.15. Paired t-Test for Mean Total Unit Cost from the MILP Method with the Means from the dcMST/MILP Combination and Heuristic Methods (10 Nodes)

	MILP	*Combo*		*Heuristic 1*		*Heuristic 2*
Mean	824.24	861.02		887.33		869.00
Variance	3172.64	5289.60		3730.84		7490.89
Observations	10	10		10		10
Hypothesized Mean Difference	0			0		0
df	9			9		9
t Stat	-2.187			-6.258		-2.257
P(T<=t) one-tail	0.028			0.000		0.025
t Critical one-tail	1.833			1.833		1.833
P(T<=t) two-tail	0.057			0.000		0.050
t Critical two-tail	2.262			2.262		2.262

The p-values in Table 4.15 indicate the solution to the MILP formulation dominates the solutions of the combination and heuristic strategies for the 10-node instance of the problem.

For the instances with 15 or more nodes, the percentages of the commodities that are dropped for each method are not similar. Therefore the means cannot be fairly compared. In Figures 4.16 and 4.17, however, we show the costs from each method for the 15- and 20-node instances, respectively. On average for problems with 20 nodes, 121-132% more commodities are dropped in the heuristic solutions than in the solution from the dcMST/MILP combination strategy.

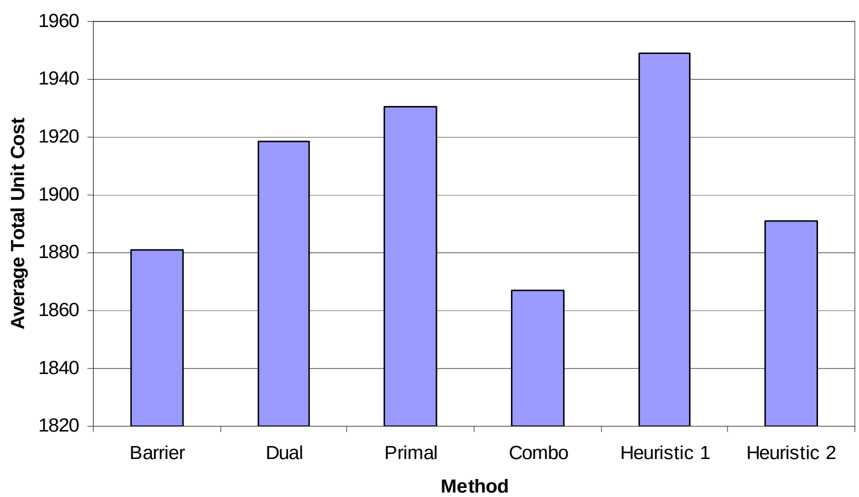

Figure 4.15. Comparison of Mean Total Unit Cost for All Methods (15 Nodes)

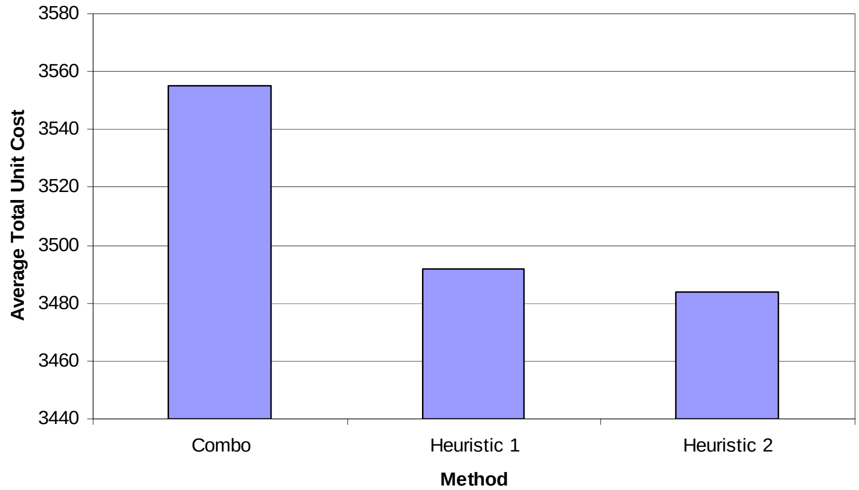

Figure 4.16. Comparison of Mean Total Unit Cost for dcMST/MILP Combination and Heuristic Methods (20 Nodes)

Other Network Topology Metrics

Other metrics used in previous studies and discussed in Chapter II include the average number of hops per commodity and the diameter of the network topology. These metrics provide a quantitative measure of a network's connectivity. The average number of hops in a topology is also indicative of the network's delay. While these metrics are not explicitly included in the objectives of any method used in this study, we consider them to evaluate indirect consequences of each topology design strategy.

Recall that the diameter of a network is the maximum hop distance between all source-destination node pairs. We present the scaling behavior of the network diameter as the problem size increases for each method in Figure 4.18.

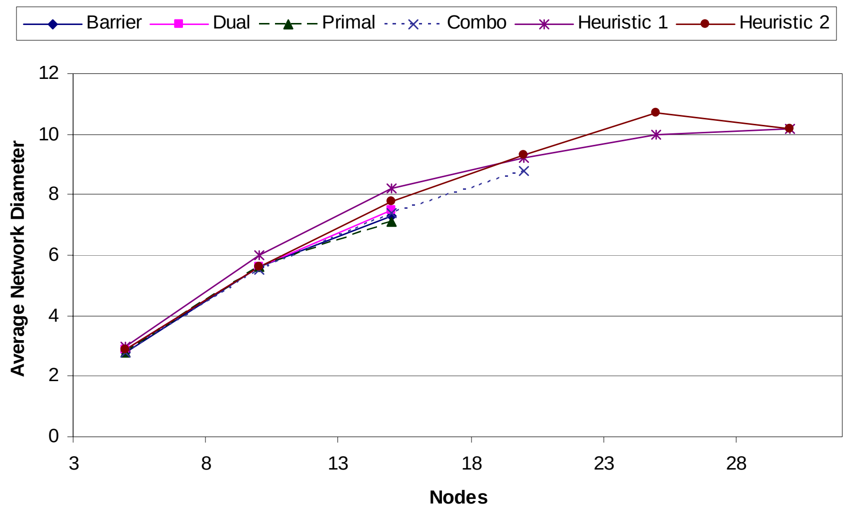

Figure 4.17. Number of Nodes vs. Network Diameter for All Methods (5-30 Nodes)

From Figure 4.18, we see no significant difference between any of the methods. Interestingly, the curve for each method appears to be concave on the given interval. Concavity is a good sign for the scalability of the network diameter with increased problem size, because lesser values are desired.

We present the scaling behavior of the average number of hops per commodity as the problem size increases for each method in Figure 4.19.

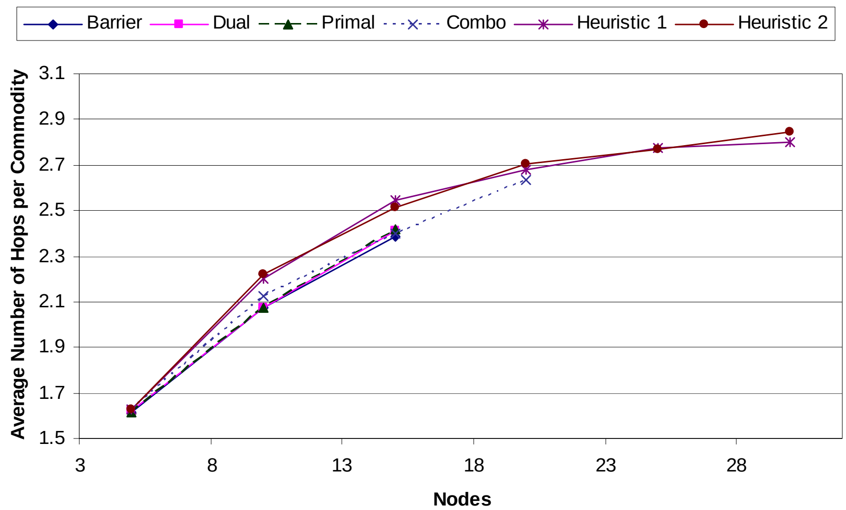

Figure 4.18. Number of Nodes vs. Average Number of Hops per Commodity for All Methods (5-30 Nodes)

Any differences between the MILP methods and the dcMST/MILP method are insignificant. There appears to be a separation between the MILP methods and the heuristic methods for problems with 10-15 nodes. Tables 4.16 and 4.17 show the statistical analysis on the differences of these means.

Table 4.16. Paired t-Tests for Average Number of Hops from the MILP Method with the Means from the Heuristic Methods (10 Nodes)

	MILP	*Heuristic 1*	*Heuristic 2*
Mean	2.072	2.203	2.222
Variance	0.011	0.019	0.020
Observations	10	10	10
Hypothesized Mean Difference	0		0
df	9		9
t Stat	-3.841		-4.974
P(T<=t) one-tail	0.002		0.000
t Critical one-tail	1.833		1.833
P(T<=t) two-tail	0.004		0.001
t Critical two-tail	2.262		2.262

Table 4.17. Paired t-Tests for Average Number of Hops from the MILP Method with the Means from the Heuristic Methods (15 Nodes)

	MILP	*Heuristic 1*		*Heuristic 2*
Mean	2.383	2.546		2.513
Variance	0.008	0.014		0.008
Observations	10	10		10
Hypothesized Mean Difference	0			0
df	9			9
t Stat	-3.629			-4.480
P(T<=t) one-tail	0.003			0.001
t Critical one-tail	1.833			1.833
P(T<=t) two-tail	0.005			0.002
t Critical two-tail	2.262			2.262

The results from the paired t-tests shown in Tables 4.16 and 4.17 show that the means from the heuristic strategies are not equal to the mean from the MILP method. Because the minimization of the average number of hops is desired, we conclude that the MILP methods dominate the heuristic strategies for instance of 10 and 15 nodes.

Recall that the post-processing for the heuristic strategies could not find a feasible integer solution within the given time limit for instances of 35 and 39 nodes, hence we do not have the data to provide the mean network diameter and average number of hops. We can, however, compare the number of links chosen in the topologies generated by the heuristic strategies as an approximating metric of connectivity.

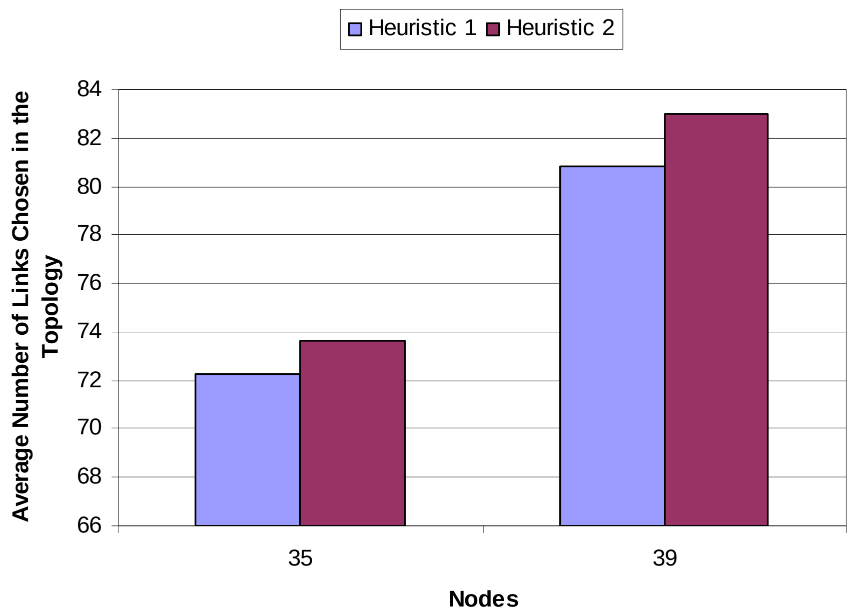

Figure 4.19. Number of Nodes vs. Average Number of Links Chosen for Both Heuristic Methods (35 and 39 Nodes)

Tables 4.18 and 4.19 show the results of the paired t-test for the means shown in Figure 4.20.

Table 4.18. Paired t-Test for Average Number of Links Chosen for Both Heuristic Methods (35 Nodes)

	Heuristic 1	*Heuristic 2*
Mean	72.3	73.6
Variance	12.233	17.822
Observations	10	10
Hypothesized Mean Difference	0	
df	9	
t Stat	-1.647	
P(T<=t) one-tail	0.067	
t Critical one-tail	1.833	
P(T<=t) two-tail	0.134	
t Critical two-tail	2.262	

With $\alpha = 0.07$, we reject the hypothesis for the one-tailed t-test, and Heuristic 1 outperforms Heuristic 2. Otherwise, we cannot infer a difference in the means.

Table 4.19. Paired t-Test for Average Number of Links Chosen for Both Heuristic Methods (39 Nodes)

	Heuristic 1	*Heuristic 2*
Mean	80.8	83
Variance	28.178	27.111
Observations	10	10
Hypothesized Mean Difference	0	
df	9	
t Stat	-3.404	
P(T<=t) one-tail	0.004	
t Critical one-tail	1.833	
P(T<=t) two-tail	0.008	
t Critical two-tail	2.262	

For the 39-node instance, we can draw a stronger conclusion. With $\alpha = 0.01$, we reject the hypothesis. Assuming a better topology has more links, we find that Heuristic 2 dominates Heuristic 1.

Tradeoff Comparisons

As previously stated, there are tradeoffs between certain metrics. A less expensive network topology might require the omission of more commodities. Another topology may have a greater cost, but requires less time to identify. The decision of how much of a tradeoff is acceptable must be addressed by the network users and/or administrators.An examination of the tradeoffs associated with the solution strategies in this research is presented in hopes of providing insight into such a decision.

Figure 4.21 shows the tradeoff between computational time and solution cost for the 5-node instance.

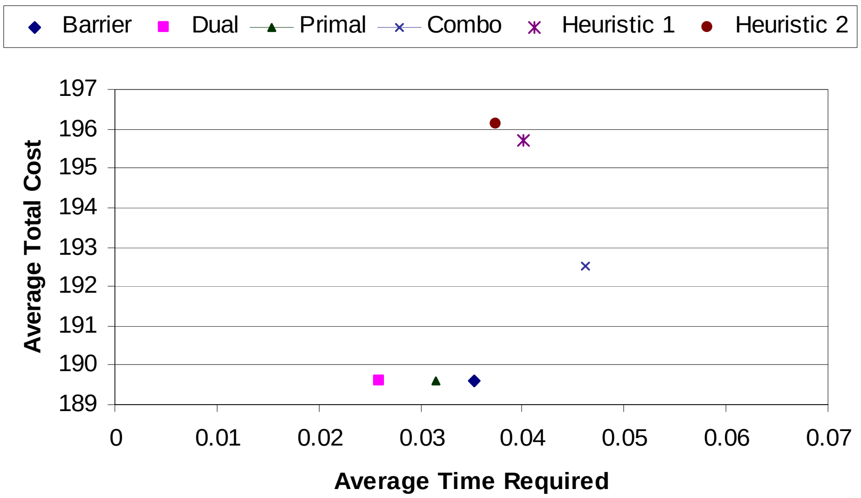

Figure 4.20. Average Time Required vs. Average Total Cost Tradeoff Comparison (5 Nodes)

Because the minimization of both of these metrics is desired, we see that the MILP methods dominate the other solution methods with less required time and lower total cost. As shown in Figure 4.22, this trend does not continue as the size of the problem increases to 10 nodes.

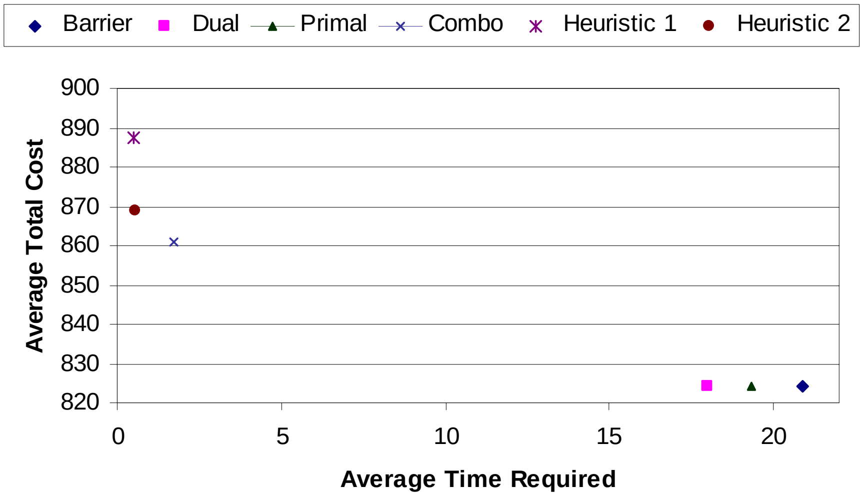

Figure 4.21. Average Time Required vs. Average Total Cost Tradeoff Comparison (10 Nodes)

With 10 nodes, the MILP methods provide better solutions, but require more computational time. With more than 10 nodes, the comparison becomes untenable due to the difference in the average number of dropped commodities.

Summary

This chapter discussed the parameters used in the generation of data sets for the purpose of testing. Due to the nature of the network requirements, we are unable to generate data sets with more than 39 nodes. Using the metrics discussed in Chapter II, we compared the solutions found by each solution strategy. The metrics used include computational complexity, number of dropped commodities, solution quality, average number of hops per commodity, and network diameter. The tradeoffs between certain metrics associated with each strategy were also examined. This chapter provides the

results and analysis to develop the conclusions and further research suggestions discussed in Chapter V.

V. Conclusions and Recommendations

Overview

This research explored a unique instance of the NDP. In Chapter I we introduced the problem considered in this study along with its background, our approach to the problem, and the scope of our research. In Chapter II, we discussed communications networks and the many instances of the NDP with approaches employed in previous studies. We also discussed different solution methods and various metrics used in past studies. In Chapter III, we covered the intricacies of the instance of the NDP considered in our research. We discussed the network characteristics and requirements that would need to be generated. We also presented the several solution strategies which we would use to solve the NDP. In Chapter IV, we discussed the results of generating network data sets for testing our solution methods. We also presented the results of our tests and comparisons using the metrics discussed in Chapter II. In this chapter, we draw conclusions from the results and comparisons presented in the previous chapter. We also provide suggestions for further research in the area of topology control in directional hybrid wireless networks.

Conclusions

In Chapter III, we stated our assumptions, which were made with the intent of outlining a broad, general definition of the problem. We defined the problem in this way to provide a bound for the performance of each of the solution methods considered in our study. We know, therefore, that the performance of any of our solution methods will be

no worse for any real-world application where potentially many simplifying assumptions can be made.

Our analysis shows that the MILP method does not scale well with an increase in problem size. As problem size increased, on average, the Newton Barrier and Dual Simplex methods both dominated the Primal Simplex method, and the Newton Barrier method also dominated the Dual Simplex method for problems with 15 nodes in terms of computational time. Beginning the MILP approach with a partial solution such as the dcMST enables the solution of larger problems but eventually encounters similar scaling issues. The heuristic solution methods enabled us to solve much larger problems but the post-processing MILP portion eventually fell victim to the scaling problem experienced with the other methods. Without post-processing, we were limited only by the inability to create data sets for larger problem sizes. On average, our analysis shows Heuristic 1 constructs a topology more quickly than Heuristic 2.

Our analysis shows no significant difference in the number of dropped commodities for the MILP methods. For the 10-node instance, the only discernable difference between the means for this metric lies between the MILP methods and Heuristic 2. As the problem size increased, the MILP and combination methods greatly outperformed the heuristic strategies, and the MILP methods failed to dominate the combination method.

The Newton Barrier method also dominated the Dual and Primal Simplex methods in terms of solution quality. The MILP methods dominated the other three methods for the problems with 5 and 10 nodes. We cannot fairly compare the costs for problems with more than 10 nodes, because of the disparity between the numbers of

dropped commodities which distorts the comparison. For the problem with 20 nodes, the heuristic strategies provided solutions comparable to those of the dcMST/MILP combination method, but drop more than twice as many commodities than the combination method.

The analysis of our research shows no significant difference in the average network diameter between any of the methods for any of the problem sizes. In regards to the average number of hops per commodity, the results from our analysis show that the MILP methods are dominated. Interestingly, over the range of problems tested in this study, both the network diameter and the average number of hops scale well with increased problem size. Also, as problem size increases, the number of links included in the topology by Heuristic 1 is less than the number included by Heuristic 2. This difference provides some insight into the behavior of the heuristic methods in the absence of post-processing.

Recommendations for Future Research

In this research, we assumed that the node-incidence matrix was common to all of the commodities. In reality, this may not be the case, and such cases should be examined in more detail. For example, hazardous materials cannot travel through urban centers. They must be transported on a route that circumvents populated areas, while non-hazardous material can proceed on a more direct routing. As another example, note that secured communications cannot be sent through unsecured lines, while unsecured communications can be sent through any line. In these examples, we clearly see that

links available for one commodity may not necessarily be available for other commodities.

In our heuristic solution strategies, we begin by constructing a dcMST backbone and then added links to form a mesh topology. When constructing a backbone network, Cahn [19] reports the network designer must first determine whether there are natural traffic centers in the network, or if all nodes have similar traffic. He defines "big" nodes as ones with many interfaces and "small" nodes as ones with few interfaces. While it may be acceptable for small nodes to route their traffic via big nodes, routing traffic between the big nodes via the small nodes was deemed undesirable. We suggest the investigation of a method or rule to determine which nodes to consider for the backbone. Possible nodes to consider may include ones where high amounts of traffic will flow, ones that have superior interface types, or ones that have a relatively large number of interfaces. Once the backbone is built, another heuristic or MILP could then be used to design an access network.

The type of telecommunications network considered in this research is one intended to serve users in dynamic environment. At any given time, a characteristic or requirement of the network may change. We suggest the investigation of possible methods to handle disruptions in the network. Possible disruptions include node additions or deletions, link additions or deletions, changes in traffic requirements, and changes in link or flow costs. Future research should explore methodologies to update the network topology with minimal disruption to the network. This should include investigating the tradeoffs associated with computational times necessary to obtain a new

topology, versus minimizing disruptions to traffic on the current topology, particularly at key areas in the network.

We considered two main objectives in our study, namely, total cost and commodity demand satisfaction. We searched for a solution that would satisfy as many commodity demands as possible at a minimum total cost. In further research, one might consider adding other objectives such as network diameter, throughput, average number of hops, reliability, computational time, and delay. Perhaps Value Focused Thinking or Goal Programming techniques could be utilized to incorporate all objectives into a single objective function.

We suggest for further research the manipulation of our solution strategies to improve efficiency or the development and evaluation of entirely different strategies. The MILP formulation has serious scaling problems that could possibly be mitigated through the use of a decomposition method. Various methods including Lagrangian Relaxation and Dantzig-Wolfe Decomposition can be considered. Future research could evaluate the usefulness of the different decomposition methods for this type of MILP formulation. One could also examine the use of different heuristic strategies for adding or swapping links. Another suggestion for mitigating the scalability of the MILP formulations is to eliminate the MILP formulations from the solution strategy. Instead, the researcher can develop a new solution method, possibly using meta-heuristics.

We handled the omission of commodities with a prioritization based on the bandwidth requirement of each commodity. We also showed the disadvantage of using a pre-emptive ordering of the commodities according to their priority. Other approaches for the prioritization of the commodities must be explored. Perhaps each commodity

could be assigned a discrete value indicating its priority level. One could also research how commodity traffic flow and insufficient bandwidth is handled currently in telecommunications networks and model the decision to drop commodities after existing practices.

Finally, and perhaps most importantly, we suggest simplifying the requirements for the network characteristics data set. We used arbitrary values to maintain generality, but a few key assumptions could possibly reduce the complexity of the problem by a significant amount, which would allow for the testing of larger problems. For instance, link cost might depend on interface type. Perhaps a link from interface type A always costs 10 units, regardless of which nodes it connects. Flow cost could depend only on the commodity (maybe its priority or type) and the link type. For instance, commodity 1 flowing on a link from interface type A could cost 5 units per unit flow. Similarly, capacities of links would depend merely on link type. Also, in our research, we assume the existence of a commodity for every source-destination node pair. This assumption provided us with a "worst case" scenario for the commodity traffic demands in a network. An appropriate number of commodities for a realistic communications network scenario should be determined and implemented in further research.

Appendix A: Test Results

5 Nodes

Method	Trial	link cost	flow cost	total cost	avg # hops	diameter	dropped comm	top time	flow time	total
LP Barrier	1	51	163	214	1.7	3	0			0.033419
	2	46	135	181	1.6	3	0			0.028396
	3	54	142	196	1.55	2	0			0.0354
	4	30	178	208	1.8	3	0			0.018122
	5	50	151	201	1.65	3	0			0.032447
	6	41	127	168	1.6	3	0			0.031745
	7	55	116	171	1.6	3	0			0.031713
	8	40	136	176	1.5	2	0			0.04647
	9	52	155	207	1.6	3	0			0.065056
	10	38	136	174	1.55	3	0			0.029511
	avg	45.7	143.9	189.6	1.615	2.8	0			0.0352279
	std dev	8.179242	18.174769	17.392208	0.08514693	0.421637	0			0.0125768
LP Dual	1	51	163	214	1.7	3	0			0.024118
	2	46	135	181	1.6	3	0			0.020999
	3	54	142	196	1.6	3	0			0.022486
	4	30	178	208	1.8	3	0			0.019382
	5	50	151	201	1.65	3	0			0.022488
	6	41	127	168	1.6	3	0			0.015418
	7	55	116	171	1.6	3	0			0.021286
	8	40	136	176	1.5	2	0			0.041748
	9	52	155	207	1.6	3	0			0.057119
	10	38	136	174	1.55	3	0			0.013804
	avg	45.7	143.9	189.6	1.62	2.9	0			0.0258848
	std dev	8.179242	18.174769	17.392208	0.08232726	0.316228	0			0.0133176
LP Primal	1	51	163	214	1.7	3	0			0.028746
	2	46	135	181	1.6	3	0			0.023056
	3	54	142	196	1.55	2	0			0.02572
	4	30	178	208	1.8	3	0			0.020761
	5	50	151	201	1.65	3	0			0.028968
	6	41	127	168	1.6	3	0			0.020328
	7	55	116	171	1.6	3	0			0.029839
	8	40	136	176	1.5	2	0			0.05115
	9	52	155	207	1.6	3	0			0.062645
	10	38	136	174	1.55	3	0			0.023398
	avg	45.7	143.9	189.6	1.615	2.8	0			0.0314611
	std dev	8.179242	18.174769	17.392208	0.08514693	0.421637	0			0.0140766
Combo	1	51	163	214	1.7	3	0			0.07282
	2	46	135	181	1.6	3	0			0.059416
	3	54	142	196	1.65	3	0			0.05772
	4	30	178	208	1.8	3	0			0.028057
	5	50	151	201	1.65	3	0			0.046254
	6	41	127	168	1.6	3	0			0.036301
	7	65	115	180	1.65	3	0			0.041184
	8	37	143	180	1.5	2	0			0.03311
	9	45	169	214	1.6	2	0			0.047201
	10	49	134	183	1.5	3	0			0.039921
	avg	46.8	145.7	192.5	1.625	2.8	0			0.0461984
	std dev	9.6124919	19.658473	16.277114	0.08897565	0.421637	0			0.013663
Heuristic 1	1	62	156	218	1.7	3	0	0.004733	0.028751	0.033484
	2	60	124	184	1.6	3	0	0.01923	0.039272	0.058502
	3	68	133	201	1.55	3	0	0.018968	0.036223	0.055191
	4	42	176	218	1.8	3	0	0.004091	0.008189	0.01228
	5	66	139	205	1.65	3	0	0.017809	0.02144	0.039249
	6	56	118	174	1.6	3	0	0.004959	0.049715	0.054674
	7	65	115	180	1.7	3	0	0.004948	0.033009	0.037957
	8	37	143	180	1.55	3	0	0.004937	0.023045	0.027982
	9	45	169	214	1.65	3	0	0.004929	0.031188	0.036117
	10	49	134	183	1.5	3	0	0.004672	0.04039	0.045062
	avg	55	140.7	195.7	1.63	3	0	0.0089276	0.0311222	0.0400498
	std dev	11.025224	20.677685	17.340063	0.08881942	0	0	0.0067365	0.0116339	0.0140993
Heuristic 2	1	62	156	218	1.7	3	0	0.004738	0.029868	0.034606
	2	60	124	184	1.6	3	0	0.012423	0.036371	0.048794
	3	68	133	201	1.55	3	0	0.011489	0.030474	0.041963
	4	42	176	218	1.8	3	0	0.004376	0.008244	0.01262
	5	66	139	205	1.65	3	0	0.018383	0.021746	0.040129
	6	56	118	174	1.6	3	0	0.004908	0.047747	0.052655
	7	65	115	180	1.7	3	0	0.005075	0.033527	0.038602
	8	37	143	180	1.55	3	0	0.004924	0.016328	0.021252
	9	47	171	218	1.6	2	0	0.004924	0.031906	0.03683
	10	49	134	183	1.5	3	0	0.004818	0.041009	0.045827
	avg	55.2	140.9	196.1	1.625	2.9	0	0.0076058	0.029722	0.0373278
	std dev	10.840254	20.989151	17.847813	0.08897565	0.316228	0	0.0048181	0.011646	0.01223

10 Nodes

Method	Trial	link	flow	total	avg # hops	diameter	dropped comm	top time	flow time	total
LP Barrier	1	199	616.9	815.9	2.07778	5	0			29.2335
	2	160	663.45	823.45	2.12	5	0			13.12
	3	179	584	763	1.96	5	0			15.382
	4	138	760	898	2.26	5	0			11.705
	5	169	597.8	766.8	2.04	6	0			23.59
	6	190	623.42	813.42	2.04	8	0			16
	7	168	568.8	736.8	1.92	4	0			9.591
	8	119	760.65	879.65	2.2	6	0			72.21
	9	133	724.33	857.33	2.09	7	0			10.16
	10	158	730	888	2.01	5	0			7.939
	avg	161.3	662.935	824.235	2.071778	5.6	0			20.89305
	std dev	25.403631	74.723578	56.326221	0.10324131	1.173788	0			19.20695
LP Dual	1	199	616.9	815.9	2.07778	5	0			25.3983
	2	160	663.45	823.45	2.12	5	0			11.21
	3	179	584	763	1.96	5	0			16.624
	4	138	760	898	2.26	5	0			11.944
	5	169	597.8	766.8	2.04	6	0			29.8
	6	190	623.42	813.42	2.04	8	0			14.883
	7	168	568.8	736.8	1.92	4	0			7.657
	8	119	760.65	879.65	2.2	6	0			45.48
	9	133	724.33	857.33	2.09	7	0			7.551
	10	158	730	888	2.01	5	0			9.118
	avg	161.3	662.935	824.235	2.071778	5.6	0			17.96653
	std dev	25.403631	74.723578	56.326221	0.10324131	1.173788	0			12.195407
LP Primal	1	199	616.9	815.9	2.07778	5	0			22.8829
	2	160	663.45	823.45	2.12	5	0			13.11
	3	179	584	763	1.96	5	0			24.29
	4	138	760	898	2.26	5	0			11.693
	5	169	597.8	766.8	2.04	6	0			23.22
	6	190	623.42	813.42	2.04	8	0			17.53
	7	168	568.8	736.8	1.92	4	0			8.754
	8	119	760.65	879.65	2.2	6	0			49.9
	9	133	724.33	857.33	2.09	7	0			8.114
	10	158	730	888	2.01	5	0			13.631
	avg	161.3	662.935	824.235	2.071778	5.6	0			19.31249
	std dev	25.403631	74.723578	56.326221	0.10324131	1.173788	0			12.280336
Combo	1	157	706	863	2.0222	6	0			2.42545
	2	161	759.6	920.6	2.34444	7	0			2.13579
	3	136	652.6	788.6	2.04	4	0			1.323
	4	120	812	932	2.27	6	0			1.26
	5	139	678.25	817.25	2.0444	6	0			0.948
	6	163	631.8	794.8	1.98	5	3			1.541
	7	151	624	775	1.98	6	0			3.422
	8	136	817.33	953.33	2.27	6	0			1.47
	9	110	845.6	955.6	2.17	5	0			0.8
	10	123	687	810	2.1	4	10			1.839
	avg	139.6	721.418	861.018	2.122104	5.5	1.3			1.716424
	std dev	18.258636	81.531592	72.729643	0.13296167	0.971825	3.198958164			0.780848
Heuristic 1	1	142	773.333	915.333	2.22222	7	0	0.086185	0.17342	0.259605
	2	176	753.6	929.6	2.43	7	0	0.118	0.265	0.383
	3	140	657.5	797.5	2.01	6	0	0.033	0.791	0.824
	4	120	812	932	2.29	6	0	0.134	0.799	0.933
	5	139	678.25	817.25	2.08	6	0	0.014	0.188	0.202
	6	189	676.05	865.05	2.13	6	3	0.026	0.351	0.377
	7	161	646.6	807.6	2.04	6	0	0.081	0.511	0.592
	8	136	817.33	953.33	2.29	6	0	0.012	0.422	0.434
	9	110	845.6	955.6	2.19	5	0	0.17	0.436	0.606
	10	132	768	900	2.35	5	9	0.032	0.194	0.226
	avg	144.5	742.8263	887.3263	2.203222	6	1.2	0.0706185	0.413042	0.4836605
	std dev	24.313919	72.837668	61.080601	0.13868025	0.666667	2.898275349	0.0558053	0.2318588	0.2504083
Heuristic 2	1	154	726.333	880.333	2.24444	6	0	0.079733	0.188961	0.268694
	2	189	616.2	805.2	2.21	5	6	0.105	1.019	1.124
	3	145	662	807	2.23	5	8	0.033	0.412	0.445
	4	107	836.5	943.5	2.3	6	0	0.135	0.614	0.749
	5	139	678.25	817.25	2.08	6	0	0.009	0.175	0.184
	6	189	669.37	858.37	2.22	6	3	0.026	0.413	0.439
	7	147	614	761	2.01	5	3	0.085	0.543	0.628
	8	125	891.38	1016.38	2.51	8	0	0.013	0.721	0.734
	9	111	878	989	2.32	5	0	0.171	0.453	0.624
	10	133	679	812	2.1	4	10	0.031	0.193	0.224
	avg	143.9	725.1033	869.0033	2.222444	5.6	3	0.0687733	0.4731961	0.5419694
	std dev	28.136374	104.9068	86.549954	0.14116089	1.074968	3.771236166	0.0555295	0.2666675	0.2899347

15 Nodes

Method	Trial	link	flow	total	avg # hops	diameter	dropped comm	top time	flow time	total
LP Barrier	1	338	1411.07	1749.07	2.28	8	0			1299.26
	2	321	1468.5	1789.5	2.45	7	8			297.9
	3	316	1569.15	1885.15	2.32	8	3			283.49
	4	275	1586.23	1861.23	2.28	7	1			531.52
	5	305	1619.88	1924.88	2.42	8	0			1847.7
	6	294	1525.1	1819.1	2.39	6	0			1849.82
	7	283	1531.6	1814.6	2.4	6	18			474.58
	8	338	1613.4	1951.4	2.46	8	3			62.54
	9	274	1862.7	2136.7	2.54	8	15			68.75
	10	278	1602.42	1880.42	2.29	7	10			357.14
	avg	302.2	1579.005	1881.205	2.383	7.3	5.8			707.27
	std dev	25.103342	120.00033	108.87834	0.08857514	0.823273	6.629898608			693.99286
LP Dual	1	345	1390.82	1735.82	2.31	7	0			1642.77
	2	300	1591.2	1891.2	2.45	7	8			309.25
	3	319	1551.3	1870.3	2.27	8	3			278.96
	4	298	1643.87	1941.87	2.46	7	1			665.88
	5	268	1613.83	1881.83	2.42	8	0			1865.38
	6	291	1495.03	1786.03	2.29	6	0			1861.67
	7	269	1679.77	1948.77	2.39	8	18			440.32
	8	396	1594.67	1990.67	2.44	7	3			50.659
	9	257	2014.65	2271.65	2.73	10	14			83.29
	10	277	1587.97	1864.97	2.33	7	10			427.54
	avg	302	1616.311	1918.311	2.409	7.5	5.7			762.5719
	std dev	42.176876	161.64105	145.13766	0.13261892	1.080123	6.481597883			732.7284
LP Primal	1	333	1469.5	1802.5	2.28	6	0			1898.62
	2	317	1524.33	1841.33	2.41	6	7			329.73
	3	296	1849.67	2145.67	2.57	10	3			394.39
	4	265	1616.98	1881.98	2.3	6	1			388.23
	5	284	1607.53	1891.53	2.39	7	0			1974.33
	6	291	1522.77	1813.77	2.33	6	0			1966.39
	7	284	1594.1	1878.1	2.4	8	18			607.23
	8	338	1576	1914	2.41	7	3			155.55
	9	236	2048.2	2284.2	2.75	9	16			524.87
	10	274	1576.87	1850.87	2.34	6	10			439.37
	avg	291.8	1638.595	1930.395	2.418	7.1	5.8			867.871
	std dev	31.190454	175.9211	157.26647	0.14195461	1.449138	6.762642482			753.79574
Combo	1	282	1524.48	1806.48	2.24	7	1			79.27
	2	291	1529.7	1820.7	2.51	8	8			167.15
	3	257	1827.77	2084.77	2.56	10	3			131.97
	4	268	1621.3	1889.3	2.28	6	1			96.97
	5	252	1714.2	1966.2	2.36	8	0			103.06
	6	274	1537.07	1811.07	2.39	7	0			78.23
	7	223	1513.7	1736.7	2.46	8	40			16.196
	8	300	1496.85	1796.85	2.34	6	4			102.07
	9	217	1742.97	1959.97	2.54	6	19			30.07
	10	281	1517.48	1798.48	2.32	8	10			624.39
	avg	264.5	1602.552	1867.052	2.4	7.4	8.6			142.9376
	std dev	27.557012	117.9701	106.42354	0.11185308	1.264911	12.52730351			174.77589
Heuristic 1	1	262	1652.88	1914.88	2.52	9	12	0.045	4	4.045
	2	293	1647.55	1940.55	2.63	7	9	0.253	3.057	3.31
	3	250	1723.23	1973.23	2.65	9	22	0.147	4.14	4.287
	4	223	1819.35	2042.35	2.44	8	7	0.204	2.649	2.853
	5	249	2041.63	2290.63	2.65	9	4	0.054	5.48	5.534
	6	254	1824.73	2078.73	2.71	11	0	0.143	3.393	3.536
	7	213	1631.67	1844.67	2.61	6	40	0.0342	1.342	1.3762
	8	270	1649.37	1919.37	2.38	8	4	0.045	2.676	2.721
	9	195	1353.9	1548.9	2.47	8	60	0.187	0.98	1.167
	10	244	1694.9	1938.9	2.4	7	10	3.752	2.97	6.722
	avg	245.3	1703.921	1949.221	2.546	8.2	16.8	0.48642	3.0687	3.55512
	std dev	28.573686	176.04098	187.38613	0.11843423	1.398412	19.05430835	1.1499878	1.3173602	1.7133694
Heuristic 2	1	272	1574.37	1846.37	2.54	8	15	0.045	1.967	2.012
	2	278	1507.75	1785.75	2.6	7	17	0.252	3.641	3.893
	3	237	1632.67	1869.67	2.45	9	25	0.098	3.139	3.237
	4	260	1771.3	2031.3	2.48	7	1	0.202	2.052	2.254
	5	250	1870.73	2120.73	2.57	8	3	0.058	4.476	4.534
	6	260	1810.153	2070.153	2.59	8	0	0.15	3.311	3.461
	7	211	1718.77	1929.77	2.56	8	43	0.037	1.817	1.854
	8	302	1574.45	1876.45	2.4	7	4	0.044	3.452	3.496
	9	181	1376.1	1557.1	2.6	8	67	0.2	1.37	1.57
	10	252	1570.55	1822.55	2.34	8	15	3.753	6.338	10.091
	avg	250.3	1640.6843	1890.9843	2.513	7.8	19	0.4839	3.1563	3.6402
	std dev	34.360992	151.37977	161.92233	0.09129318	0.632456	21.39054828	1.1512565	1.4843604	2.4662141

20 Nodes

Method	Trial	link cost	flow cost	total cost	avg # hops	diameter	dropped comm	top time	flow time	total
Combo	1	256	3879.1	4135.1	2.7	10	10			1434.93
	2	289	2972.9	3261.9	2.6	8	68			139.51
	3	403	2905.32	3308.32	2.61	9	0			937.15
	4	373	2940.75	3313.75	2.48	8	15			511.95
	5	344	3175.53	3519.53	2.6	9	0			507.84
	6	284	3377	3661	2.67	9	26			284.74
	7	303	3259.28	3562.28	2.65	10	21			235.47
	8	329	3047.37	3376.37	2.6	8	17			910.74
	9	377	3146.55	3523.55	2.66	8	8			617.01
	10	323	3566.53	3889.53	2.76	9	9			832.665
	avg	328.1	3227.033	3555.133	2.633	8.8	17.4			641.2005
	std dev	46.665357	308.29805	278.65376	0.07469196	0.788811	19.64235786			395.15867
Heuristic 1	1	256	3134.33	3390.33	2.57	9	54	0.589	26.68	27.269
	2	276	2721.45	2997.45	2.66	9	94	0.409	40.48	40.889
	3	350	3325.02	3675.02	2.76	10	0	0.0811	11.478	11.5591
	4	372	3027.85	3399.85	2.66	9	15	0.296	12.51	12.806
	5	334	3927.55	4261.55	2.77	11	15	0.388	49.51	49.898
	6	282	3437.41	3719.41	2.68	8	34	0.091	16.823	16.914
	7	286	2801.33	3087.33	2.58	10	64	0.094	22.1	22.194
	8	317	3070.9	3387.9	2.69	7	33	1.588	14.253	15.841
	9	364	3086.8	3450.8	2.77	11	42	0.363	34.36	34.723
	10	303	3244.28	3547.28	2.67	8	34	0.091	11.85	11.941
	avg	314	3177.692	3491.692	2.681	9.2	38.5	0.39901	24.0044	24.40341
	std dev	39.869231	341.5136	352.79423	0.07125073	1.316561	27.1630877	0.4528626	13.430598	13.439287
Heuristic 2	1	266	3432.78	3698.78	2.8	9	41	0.609	27.39	27.999
	2	294	3040.02	3334.02	2.84	10	90	0.408	21.46	21.868
	3	338	3144.12	3482.12	2.59	9	29	0.19	9.486	9.676
	4	337	3398.61	3735.61	2.8	10	16	0.294	11.9	12.194
	5	342	3216.62	3558.62	2.57	10	7	0.379	12.441	12.82
	6	285	3204.59	3489.59	2.72	9	46	0.091	17.71	17.801
	7	290	3226.78	3516.78	2.7	9	42	0.094	11.377	11.471
	8	344	2930.32	3274.32	2.65	8	25	1.604	19.399	21.003
	9	335	2777	3112	2.64	10	71	0.37	25.56	25.93
	10	326	3310.77	3636.77	2.75	9	36	0.092	14.758	14.85
	avg	315.7	3168.161	3483.861	2.706	9.3	40.3	0.4131	17.1481	17.5612
	std dev	28.802199	204.70905	195.58962	0.09264028	0.674949	24.78373122	0.4512086	6.2098187	6.3887039

25 Nodes

Method	Trial	link cost	flow cost	total cost	avg # hops	diameter	dropped comm	top time	flow time	total
Heuristic 1	1	430	6386.25	6816.25	2.86	10	49	0.563	147.53	148.093
	2	369	5508.01	5877.01	2.87	10	62	0.154	72.77	72.924
	3	386	4809.69	5195.69	2.62	9	124	0.358	83.1	83.458
	4	394	3701.45	4095.45	2.62	11	213	0.52	68.66	69.18
	5	407	5277.1	5684.1	2.8	9	67	0.497	84.52	85.017
	6	500	4849.19	5349.19	2.74	13	16	0.429	38.493	38.922
	7	528	5129.18	5657.18	2.86	10	0	0.205	16.324	16.529
	8	415	4886.47	5301.47	2.75	10	140	0.908	152.799	153.707
	9	420	5843.05	6263.05	2.78	9	48	0.184	164.34	164.524
	10	402	5124.33	5526.33	2.86	9	76	10.74	513.411	524.151
	avg	425.1	5151.472	5576.572	2.776	10	79.5	1.4558	134.1947	135.6505
	std dev	50.39279	709.30885	713.75984	0.09500877	1.247219	63.43194078	3.2698064	141.98437	145.08015
Heuristic 2	1	436	5526.12	5962.12	2.84	11	43	0.568	90.23	90.798
	2	402	5027.43	5429.43	2.77	12	66	0.158	49.21	49.368
	3	449	4892.05	5341.05	2.7	11	70	0.34	93.78	94.12
	4	426	5627.35	6053.35	2.71	9	89	0.522	1886.75	1887.272
	5	444	4845.39	5289.39	2.81	10	65	0.511	61.05	61.561
	6	490	4890.2	5380.2	2.72	8	47	0.422	154.78	155.202
	7	481	5446.32	5927.32	2.86	14	45	0.206	100.6	100.806
	8	361	5309.07	5670.07	2.71	11	125	0.899	250.915	251.814
	9	425	5444.18	5869.18	2.79	11	59	0.186	103.273	103.459
	10	398	3714.43	4112.43	2.76	10	179	12.189	106.67	118.859
	avg	431.2	5072.254	5503.454	2.767	10.7	78.8	1.6001	289.7258	291.3259
	std dev	38.571146	558.89193	565.85788	0.05735852	1.636392	42.8662001	3.7272067	563.97196	563.61025

30 Nodes

Method	Trial	link cost	flow cost	total cost	avg # hops	diameter	dropped comm	top time	flow time	total
Heuristic 1	1	398	6731.92	7129.92	2.8	11	303	0.855	2015.74	2016.595
	2	396	6485.75	6881.75	2.79	10	301	1.527	2071.96	2073.487
	3	578	6447.23	7025.23	2.825	12	152	2.597	119.015	121.612
	4	539	9072.53	9611.53	2.81	11	91	1.72	646.57	648.29
	5	510	7187.78	7697.78	2.76	10	210	0.342	556.726	557.068
	6	565	8621.52	9186.52	2.84	10	89	0.96	2260.5	2261.46
	7	506	7616.5	8122.5	2.8	10	143	0.26	666.43	666.69
	8	523	6791.39	7314.39	2.76	9	229	0.174	922.02	922.194
	9	468	7047.68	7515.68	2.78	10	218	1.265	929.2	930.465
	10	431	6777.17	7208.17	2.82	9	239	0.2	815.08	815.28
	avg	491.4	7277.947	7769.347	2.7985	10.2	197.5	0.99	1100.3241	1101.3141
	std dev	65.625537	900.91406	935.02907	0.02667187	0.918937	76.94767342	0.7986953	739.60455	739.52204
Heuristic 2	1	410	6525.43	6935.43	2.83	11	275	0.958	1967.18	1968.138
	2	438	8212.06	8650.06	2.85	10	153	1.202	738.02	739.222
	3	568	7044.67	7612.67	2.86	10	142	2.737	301.21	303.947
	4	516	7275.33	7791.33	2.88	10	181	2.23	408.66	410.89
	5	537	7769.18	8306.18	2.73	11	129	0.421	513.6	514.021
	6	542	8912.28	9454.28	2.82	9	89	0.955	2313.71	2314.665
	7	493	7198.38	7691.38	2.92	11	176	0.57	614.09	614.66
	8	533	6708.78	7241.78	2.86	11	186	0.173	780.24	780.413
	9	516	6501.72	7017.72	2.8	9	244	1.04	469.15	470.19
	10	485	6271.63	6756.63	2.89	10	238	0.196	528.57	528.766
	avg	503.8	7241.946	7745.746	2.844	10.2	181.3	1.0482	863.443	864.4912
	std dev	48.798452	839.18953	849.3571	0.05316641	0.788811	57.32567778	0.8435056	692.76185	692.5954

35 Nodes

Method	Trial	link cost	# of links	time
Heuristic 1	1	529	73	2.648
	2	571	72	0.849
	3	542	69	2.407
	4	542	71	1.8
	5	520	71	1.626
	6	506	70	9.049
	7	586	73	1.051
	8	535	69	2.398
	9	560	74	1.363
	10	687	81	1.096
	avg	557.8	72.3	2.4287
	std dev	51.237573	3.4976182	2.4088444
Heuristic 2	1	549	74	2.752
	2	547	70	0.635
	3	526	70	2.418
	4	586	77	1.794
	5	569	73	1.659
	6	556	73	8.935
	7	557	71	1.042
	8	540	71	2.361
	9	546	73	1.364
	10	758	84	1.1
	avg	573.4	73.6	2.406
	std dev	66.860053	4.2216374	2.3914342

39 Nodes

Method	Trial	link cost	# of links	time
Heuristic 1	1	532	73	4.328
	2	646	78	3.491
	3	730	88	5.314
	4	527	77	1.153
	5	624	83	5.306
	6	603	82	4.277
	7	654	86	7.185
	8	615	87	5.304
	9	588	80	3.409
	10	532	74	1.37
	avg	605.1	80.8	4.1137
	std dev	64.293511	5.3082745	1.854801
Heuristic 2	1	515	74	3.701
	2	625	78	6.044
	3	711	89	3.838
	4	661	83	4.467
	5	651	86	4.222
	6	636	82	6.188
	7	666	89	6.173
	8	713	89	6.472
	9	629	81	6.375
	10	602	79	5.128
	avg	640.9	83	5.2608
	std dev	56.794855	5.2068331	1.1154398

Appendix B: Model Code

MILP

```
model "LPbarrier"
uses "mmxprs"
uses "mmsystem"

declarations
      v = 1                                ! version number
      NumNodes:  integer                   ! number of nodes
      NumInterfaces:  integer              ! number of different types
of interfaces
end-declarations

initializations from 'test15j.txt'
      NumNodes NumInterfaces
end-initializations

declarations
      N = 1..NumNodes                      ! set of nodes
      NumCommodities = NumNodes*(NumNodes-1)    ! number of commodities
      K = 1..NumCommodities                ! set of commodities
      F = 1..NumInterfaces                 ! set of interfaces
      Interfaces:  array(N,F) of integer  ! number of each type of
                                               interface at each node
      SourceDest:  array(K,1..4) of integer     ! array of source and
                                               destin nodes for ea.
                                               comm and req bandwidth
      FCost:  array(N,N,F) of real         ! fixed cost of each directed
                                               edge (i,j,f)
      A:  array(N,N,F) of integer          ! Node incidence matrix (arc
                                                     possibilities)
      Cap:  array(N,N,F) of real           ! capacities of all the links
      VarCost:  array(N,N,K,F) of real    ! cost of comm k to flow on
                                               (i,j,f)
      X:  array(N,N,K,F) of mpvar          ! pctg of comm k to flow on
                                               (i,j,f)
      Y:  array(N,N,F) of mpvar            ! binary var indicating
                                               whether or not we
                                               select arc (i,j,f)
      Hops:  array(K) of integer           ! the number of hops for each
                                               commodity
      s:  array(K) of mpvar                ! decision whether to drop
commodity k
end-declarations

initializations from 'test15j.txt'
      Interfaces SourceDest FCost A Cap VarCost
end-initializations
```

```
writeln(NumNodes, " nodes...")

! Record the start time
starttime := gettime

! initialize r
forall(i in N, k in K) r(i,k) := 0
forall(k in K) do
      r(SourceDest(k,2),k) := 1
      r(SourceDest(k,3),k) := -1
end-do

! Objective:  Total cost (sum of used arc costs)
VariableCost := sum(i,j in N, k in K, f in F | A(i,j,f)=1)
VarCost(i,j,k,f)*X(i,j,k,f)
FixedCost := sum(i,j in N, f in F | A(i,j,f)=1) FCost(i,j,f)*Y(i,j,f)
TotalCost := (VariableCost + FixedCost + sum(k in K)
1000*SourceDest(k,4)*s(k))

! Node balance constraints
forall(i in N, k in K) do
      if r(i,k) = 1 then
            sum(j in N, f in F | A(i,j,f)=1) X(i,j,k,f) - sum(j in N, f
                  in F | A(i,j,f)=1) X(j,i,k,f) = 1-s(k)
      elif r(i,k) = -1 then
            sum(j in N, f in F | A(i,j,f)=1) X(i,j,k,f) - sum(j in N, f
                  in F | A(i,j,f)=1) X(j,i,k,f)= -1+s(k)
      else
            sum(j in N, f in F | A(i,j,f)=1) X(i,j,k,f) - sum(j in N, f
                  in F | A(i,j,f)=1) X(j,i,k,f) = 0
      end-if
end-do

!forall(k in 1..(NumCommodities-1)) s(k) <= s(k+1)

! Link capacity constraints
forall(i,j in N, f in F | A(i,j,f)=1) sum(k in K)
X(i,j,k,f)*SourceDest(k,4) <=       Cap(i,j,f)

! Interface (degree) constraints
! Constrain the number of edges adjacent to a node (based on number of
      interfaces)
forall(i in N,f in F)
      sum(j in N) Y(i,j,f)  <= Interfaces(i,f)

! Forcing constraints
forall(i,j in N, k in K, f in F | A(i,j,f)=1) X(i,j,k,f) <= Y(i,j,f)

! If (i,j,f) exists, then (j,i,f) must exist
forall(i,j in N, f in F | A(i,j,f)=1) Y(i,j,f) = Y(j,i,f)

! The Y decision variables are binary
forall(i,j in N, f in F | A(i,j,f)=1) Y(i,j,f) is_binary
forall(k in K) s(k) is_binary
```

```
setparam("XPRS_verbose", true)
setparam("XPRS_MIPRELSTOP",0.07)
setparam("XPRS_MAXTIME",1800)

!Solve the problem with Newton Barrier, Dual Simplex, or Primal Simplex
minimize(XPRS_BAR,TotalCost)
!minimize(XPRS_DUAL,TotalCost)
!minimize(XPRS_PRI,TotalCost)

writeln("Total cost = ", getobjval)

! Get computation time
CompTime := (gettime-starttime)
ActualCost := getobjval - sum(k in K | getsol(s(k))=1)
1000*SourceDest(k,4)

LinkCost := sum(i,j in N, f in F | A(i,j,f)=1)
FCost(i,j,f)*getsol(Y(i,j,f))
FlowCost := sum(i,j in N, k in K, f in F | A(i,j,f)=1)
VarCost(i,j,k,f)*getsol(X(i,j,k,f))

! count number of hops for each commodity
forall(k in K) Hops(k) := 0
forall(i,j in N, k in K, f in F | getsol(X(i,j,k,f))<>0) do
      Hops(k) := Hops(k) + 1
end-do

! find the diameter (max num hops among all commodities)
diameter := 0
forall(k in K) do
      if Hops(k) > diameter then
            diameter := Hops(k)
      end-if
end-do

! calculate avg number of hops per commodity
Commodities := NumCommodities
forall(k in K | Hops(k)=0) Commodities := Commodities - 1
AvgNumHops := (sum(k in K | Hops(k)<>0) Hops(k))/Commodities

! find out how many commodity requirements were not met
Dropped_Commodities := 0
forall(k in K) do
      if getsol(s(k)) <> 0 then
            Dropped_Commodities := Dropped_Commodities + 1
      end-if
end-do

writeln("Total Cost = ",ActualCost)
writeln("Link Cost = ",LinkCost)
writeln("Flow Cost = ",FlowCost)
writeln("Avg # hops = ",AvgNumHops)
writeln("Diameter = ",diameter)
writeln("Time:  ", CompTime)
writeln("Dropped Commodities:  ",Dropped_Commodities)
```

```
! Write the results
fopen("runs.txt", F_OUTPUT+F_APPEND)
writeln("LPb, Nodes: ",NumNodes,", link: ",LinkCost,", flow:
      ",FlowCost,", avg       hops: ",AvgNumHops,", diam: ",diameter,",
      dc: ",Dropped_Commodities,",  time:  ",CompTime)
fclose(F_OUTPUT+F_APPEND)

!fopen("outputTwo.dat", F_OUTPUT)
!writeln("Here's the output.")
!forall(s,t in N | s<>t)
!     if getsol(X(s,t)) = 1 then
!           writeln("Arc (",s,",",t,")")
!     end-if
!fclose(F_OUTPUT)



end-model
```

dcMST/MILP Combination

```
model "combo"
uses "mmxprs"
uses "mmsystem"

declarations
      v = 1                                ! version number
      NumNodes:  integer                   ! number of nodes
      NumInterfaces:  integer              ! number of different types
of interfaces
end-declarations

initializations from 'test20j.txt'
      NumNodes NumInterfaces
end-initializations

declarations
      N = 1..NumNodes                      ! set of nodes
      NumCommodities = NumNodes*(NumNodes-1)    ! number of commodities
      K = 1..NumCommodities                ! set of commodities
      F = 1..NumInterfaces                 ! set of interfaces
      Interfaces:  array(N,F) of integer  ! number of each type of
                                                interface at each node
      Interface:  array(N,F) of integer   ! same as above, but updated
                                          in the heuristic portion of
                                          the code
      SourceDest:  array(K,1..4) of integer     ! array of source and
                                                destin nodes for ea.
                                                comm and req bandwidth
      FCost:  array(N,N,F) of integer     ! fixed cost of each directed
                                                edge (i,j,f)
      A:  array(N,N,F) of integer         ! Node incidence matrix (arc
                                                possibilities)
      Cap:  array(N,N,F) of real          ! capacities of all the links
      VarCost:  array(N,N,K,F) of real    ! cost of comm k to flow on
                                                (i,j,f)
      X:  array(N,N,K,F) of mpvar         ! pctg of comm k to flow on
                                                (i,j,f)
      Y:  array(N,N,F) of mpvar           ! binary var indicating
                                                whether or not we
                                                select arc (i,j,f)
      Hops:  array(K) of integer          ! the number of hops for each
                                                commodity

      link:  array(N,N,F) of real         ! binary var indicating
                                                whether or not we
                                                select arc (i,j,f)
      tedge:  array(N,N,F) of mpvar       ! binary var indicating
                                                whether or not the edge
                                                is in the mst
      Level: array(N) of mpvar            ! level value of nodes
      ub:  array(N) of integer            ! degree upper bound for each
                                                node
      NodeList:  array(N) of integer      ! list of the nodes in a
```

```
                                              specified order
      s:  array(K) of mpvar               ! decision whether to drop
                                              commodity k
end-declarations

initializations from 'test20j.txt'
      Interfaces SourceDest FCost A Cap VarCost
end-initializations

writeln(NumNodes, " nodes...")

! Record the start time
starttime := gettime

! Initialize link
forall(i,j in N, f in F) link(i,j,f) := 0

! Objective: cost of including edges in the network
Cost:= sum(i,j in N, f in F | A(i,j,f)=1)
(FCost(i,j,f)+FCost(j,i,f))*tedge(i,j,f)

! Number of connections
sum(i,j in N, f in F | A(i,j,f)=1) tedge(i,j,f) = NumNodes - 1

! Constrain the number of edges adjacent to a node (based on number of
      interfaces)
forall(i in N,f in F)
      (sum(j in N) tedge(i,j,f) + sum(j in N) tedge(j,i,f)) <=
            Interfaces(i,f)

! Avoid subcycle
forall(i,j in N | sum(f in F) A(i,j,f) <> 0)
Level(j) >= Level(i) + 1 - NumNodes + NumNodes*(sum(f in F)
      tedge(i,j,f))

! Direct all connections towards the root (node 1)
forall(i in 2..NumNodes) sum(j in N, f in F | A(i,j,f)=1)
      tedge(i,j,f) = 1

forall(i,j in N, f in F) tedge(i,j,f) is_binary

! Solve the problem
minimize(Cost)

! Add the mst to the network topology
forall(i,j in N, f in F | getsol(tedge(i,j,f))=1) do
      link(i,j,f) := 1
      link(j,i,f) := 1
end-do

writeln("Tree found!")

!-----------------------------------------------------------------------
!-----------------------------------------------------------------------
```

```
! initialize r
forall(i in N, k in K) r(i,k) := 0
forall(k in K) do
      r(SourceDest(k,2),k) := 1
      r(SourceDest(k,3),k) := -1
end-do

! Objective:  Total cost (sum of used arc costs)
VariableCost := sum(i,j in N, k in K, f in F | A(i,j,f)=1)
VarCost(i,j,k,f)*X(i,j,k,f)
FixedCost := sum(i,j in N, f in F | A(i,j,f)=1) FCost(i,j,f)*Y(i,j,f)
TotalCost := (VariableCost + FixedCost + sum(k in K)
1000*SourceDest(k,4)*s(k))

! keep the mst
forall(i,j in N, f in F | link(i,j,f)=1) Y(i,j,f) = 1

! Node balance constraints
forall(i in N, k in K) do
      if r(i,k) = 1 then
            sum(j in N, f in F | A(i,j,f)=1) X(i,j,k,f) - sum(j in N, f
                  in F | A(i,j,f)=1) X(j,i,k,f) = 1-s(k)
      elif r(i,k) = -1 then
            sum(j in N, f in F | A(i,j,f)=1) X(i,j,k,f) - sum(j in N, f
                  in F | A(i,j,f)=1) X(j,i,k,f)= -1+s(k)
      else
            sum(j in N, f in F | A(i,j,f)=1) X(i,j,k,f) - sum(j in N, f
                  in F | A(i,j,f)=1) X(j,i,k,f) = 0
      end-if
end-do

!forall(k in 1..(NumCommodities-1)) s(k) <= s(k+1)

! Link capacity constraints
forall(i,j in N, f in F | A(i,j,f)=1) sum(k in K)
X(i,j,k,f)*SourceDest(k,4) <= Cap(i,j,f)

! Interface (degree) constraints
! Constrain the number of edges adjacent to a node (based on number of
            interfaces)
forall(i in N,f in F)
      sum(j in N) Y(i,j,f)  <= Interfaces(i,f)

! Forcing constraints
forall(i,j in N, k in K, f in F | A(i,j,f)=1) X(i,j,k,f) <= Y(i,j,f)

! If (i,j,f) exists, then (j,i,f) must exist
forall(i,j in N, f in F | A(i,j,f)=1) Y(i,j,f) = Y(j,i,f)

! The Y decision variables are binary
forall(i,j in N, f in F | A(i,j,f)=1) Y(i,j,f) is_binary
forall(k in K) s(k) is_binary

setparam("XPRS_verbose", true)
setparam("XPRS_MIPRELSTOP",0.1)
```

```
setparam("XPRS_MAXTIME",1800)
!Solve the problem
minimize(TotalCost)

! Get computation time
CompTime := (gettime-starttime)

ActualCost := getobjval - sum(k in K | getsol(s(k))=1)
1000*SourceDest(k,4)

writeln("Total cost = ", ActualCost)

LinkCost := sum(i,j in N, f in F | A(i,j,f)=1)
FCost(i,j,f)*getsol(Y(i,j,f))
FlowCost := sum(i,j in N, k in K, f in F | A(i,j,f)=1)
VarCost(i,j,k,f)*getsol(X(i,j,k,f))

! count number of hops for each commodity
forall(k in K) Hops(k) := 0
forall(i,j in N, k in K, f in F | getsol(X(i,j,k,f))<>0) do
      Hops(k) := Hops(k) + 1
end-do

! find the diameter (max num hops among all commodities)
diameter := 0
forall(k in K) do
      if Hops(k) > diameter then
            diameter := Hops(k)
      end-if
end-do

! calculate avg number of hops per commodity
Commodities := NumCommodities
forall(k in K | Hops(k)=0) Commodities := Commodities - 1
AvgNumHops := (sum(k in K | Hops(k)<>0) Hops(k))/Commodities

! find out how many commodity requirements were not met
Dropped_Commodities := 0
forall(k in K) do
      if getsol(s(k)) <> 0 then
            Dropped_Commodities := Dropped_Commodities + 1
      end-if
end-do

writeln("Total Cost = ",ActualCost)
writeln("Link Cost = ",LinkCost)
writeln("Flow Cost = ",FlowCost)
writeln("Avg # hops = ",AvgNumHops)
writeln("Diameter = ",diameter)
writeln("Time:  ", CompTime)
writeln("Dropped Commodities:  ",Dropped_Commodities)

! Write the results
fopen("runs.txt", F_OUTPUT+F_APPEND)
writeln("Combo, Nodes: ",NumNodes,", link: ",LinkCost,", flow:
```

```
      ",FlowCost,", avg hops: ",AvgNumHops,", diam: ",diameter,", dc:
      ",Dropped_Commodities,", time:  ",CompTime)
fclose(F_OUTPUT+F_APPEND)

end-model
```

<u>Heuristic</u>

```
model "msth"
uses "mmxprs"
uses "mmsystem"

declarations
      v = 1                                ! version number
      NumNodes:  integer                   ! number of nodes
      NumInterfaces:  integer              ! number of different types
of interfaces
end-declarations

initializations from '30j.txt'
      NumNodes NumInterfaces
end-initializations

declarations
      N = 1..NumNodes                      ! set of nodes
      NumCommodities = NumNodes*(NumNodes-1)    ! number of commodities
      K = 1..NumCommodities                ! set of commodities
      F = 1..NumInterfaces                 ! set of interfaces
      Interfaces:  array(N,F) of integer   ! number of each type of
                                                interface at each node
SourceDest:  array(K,1..4) of integer      ! array of source and destin
                                                nodes for ea. comm and
                                                req bandwidth
      FCost:  array(N,N,F) of real         ! fixed cost of each directed
                                                edge(i,j,f)
      A:  array(N,N,F) of integer          ! Node incidence matrix (arc
                                                possibilities)
      Cap:  array(N,N,F) of real           ! capacities of all the links
      VarCost:  array(N,N,K,F) of real     ! cost of comm k to flow on
                                                (i,j,f)
      X:  array(N,N,K,F) of mpvar          ! pctg of comm k to flow on
                                                (i,j,f)
      Y:  array(N,N,F) of integer          ! binary var indicating
                                                whether or not we
                                                select arc (i,j,f)
      tedge:  array(N,N,F) of mpvar        ! binary var indicating
                                                whether or not the edge
                                                is in the mst
      Level: array(N) of mpvar             ! level value of nodes
      ub:  array(N) of integer             ! degree upper bound for each
                                                node
      NodeList:  array(N) of integer       ! list of the nodes in a
                                                specified order
      Hops:  array(K) of integer           ! the number of hops for each
                                                      commodity
      s:  array(K) of mpvar                ! decision whether to drop
                                                commodity k
end-declarations

initializations from '30j.txt'
      Interfaces SourceDest FCost A Cap VarCost
```

```
end-initializations

! Record the start time
starttime := gettime

! Initialize Y
forall(i,j in N, f in F) Y(i,j,f) := 0

! Objective: cost of including edges in the network
Cost:= sum(i,j in N, f in F | A(i,j,f)=1)
      (FCost(i,j,f)+FCost(j,i,f))*tedge(i,j,f)

! Number of connections
sum(i,j in N, f in F | A(i,j,f)=1) tedge(i,j,f) = NumNodes - 1

! Constrain the number of edges adjacent to a node (based on number of
      interfaces)
forall(i in N,f in F)
      (sum(j in N) tedge(i,j,f) + sum(j in N) tedge(j,i,f)) <=
Interfaces(i,f)

! Avoid subcycle
forall(i,j in N | sum(f in F) A(i,j,f) <> 0)
Level(j) >= Level(i) + 1 - NumNodes + NumNodes*(sum(f in F)
tedge(i,j,f))

! Direct all connections towards the root (node 1)
forall(i in 2..NumNodes) sum(j in N, f in F | A(i,j,f)=1)
      tedge(i,j,f) = 1

forall(i,j in N, f in F) tedge(i,j,f) is_binary

! Solve the problem
minimize(Cost)

! Add the mst to the network topology
forall(i,j in N, f in F | getsol(tedge(i,j,f))=1) do
      Y(i,j,f) := 1
      Y(j,i,f) := 1
end-do

! Update the Interfaces array to reflect the remaining available
interfaces
forall(i,j in N, f in F | Y(i,j,f) = 1) do
      Interfaces(i,f) := Interfaces(i,f) - 1
end-do

! Determine degree upper bound for all nodes
forall(i in N) ub(i) := sum(f in F) Interfaces(i,f)

! Sort the nodes in non-decreasing or non-increasing ub order
qsort(SYS_UP,ub,NodeList)
!qsort(SYS_DOWN,ub,NodeList)

! Scan through the list, adding edges as possible
```

```
count := 1
repeat
      forall(f in F) do
            ii := 1
            while ((Interfaces(NodeList(count),f) > 0) and (ii <=
                  NumNodes)) do
                  if ((A(NodeList(count),ii,f) = 1) and
                        (Y(NodeList(count),ii,f) = 0)) then
                        if Interfaces(ii,f) > 0 then
                              Y(NodeList(count),ii,f) := 1
                              Y(ii,NodeList(count),f) := 1
                              Interfaces(NodeList(count),f) :=
                                          Interfaces(NodeList(count),f)
                                                - 1
                              Interfaces(ii,f) := Interfaces(ii,f) - 1
                        end-if
                  end-if
                  ii := ii + 1
            end-do
      end-do
      count := count + 1
until (count = NumNodes)

! Get computation time
CompTime := (gettime-starttime)

NetCost := sum(i,j in N, f in F | Y(i,j,f) = 1) FCost(i,j,f)
writeln("Cost:  ",NetCost)
writeln("Time:  ",CompTime,"s.")
!-----------------------------------------------------------------------
!-----------------------------------------------------------------------

t0 := gettime

! initialize r
forall(i in N, k in K) r(i,k) := 0
forall(k in K) do
      r(SourceDest(k,2),k) := 1
      r(SourceDest(k,3),k) := -1
end-do

! Objective:  Total cost (sum of used arc costs)
VariableCost := sum(i,j in N, k in K, f in F | Y(i,j,f)=1)
VarCost(i,j,k,f)*X(i,j,k,f)
FixedCost := NetCost
TotalCost := (VariableCost + FixedCost + sum(k in K)
100000*SourceDest(k,4)*s(k))

! Node balance constraints
forall(i in N, k in K) do
      if r(i,k) = 1 then
            sum(j in N, f in F | Y(i,j,f)=1) X(i,j,k,f) - sum(j in N, f
                  in F | Y(i,j,f)=1) X(j,i,k,f) = 1-s(k)
      elif r(i,k) = -1 then
            sum(j in N, f in F | Y(i,j,f)=1) X(i,j,k,f) - sum(j in N, f
```

```
            in F | Y(i,j,f)=1) X(j,i,k,f)= -1+s(k)
    else
        sum(j in N, f in F | Y(i,j,f)=1) X(i,j,k,f) - sum(j in N, f
            in F | Y(i,j,f)=1) X(j,i,k,f) = 0
    end-if
end-do

!forall(k in 1..(NumCommodities-1)) s(k) <= s(k+1)

! Link capacity constraints
forall(i,j in N, f in F | Y(i,j,f)=1) sum(k in K)
X(i,j,k,f)*SourceDest(k,4) <=       Cap(i,j,f)

forall(k in K) s(k) is_binary

setparam("XPRS_verbose", true)
!setparam("XPRS_MIPRELSTOP",0.01)
setparam("XPRS_MAXTIME",1800)
!Solve the problem
minimize(TotalCost)
tf := gettime - t0
ActualCost := getobjval - sum(k in K | getsol(s(k))=1)
100000*SourceDest(k,4)

FlowCost := sum(i,j in N, k in K, f in F | Y(i,j,f)=1)
VarCost(i,j,k,f)*getsol(X(i,j,k,f))


! count number of hops for each commodity
forall(k in K) Hops(k) := 0
forall(i,j in N, k in K, f in F | getsol(X(i,j,k,f))<>0) do
    Hops(k) := Hops(k) + 1
end-do

! find the diameter (max num hops among all commodities)
diameter := 0
forall(k in K) do
    if Hops(k) > diameter then
        diameter := Hops(k)
    end-if
end-do

! calculate avg number of hops per commodity
Commodities := NumCommodities
forall(k in K | Hops(k)=0) Commodities := Commodities - 1
AvgNumHops := (sum(k in K | Hops(k)<>0) Hops(k))/Commodities

! find out how many commodity requirements were not met
Dropped_Commodities := 0
forall(k in K) do
    if getsol(s(k)) <> 0 then
        Dropped_Commodities := Dropped_Commodities + 1
    end-if
end-do
```

```
writeln("Total cost = ",ActualCost)
writeln("Link cost = ",FixedCost)
writeln("Flow cost = ",FlowCost)
writeln("Avg # hops = ",AvgNumHops)
writeln("Diameter = ",diameter)
writeln("Topology time:  ",CompTime,"s.")
writeln("Flows time:  ",tf,"s.")
writeln("Dropped Commodities:  ",Dropped_Commodities)

! Write the results
fopen("runs.txt", F_OUTPUT+F_APPEND)
writeln("msth1, Nodes: ",NumNodes,", link: ",FixedCost,", flow:
      ",FlowCost,",     avg hops: ",AvgNumHops,", diam: ",diameter,",
      dc: ",Dropped_Commodities,",  time:  ",CompTime,", ",tf)
fclose(F_OUTPUT+F_APPEND)

end-model
```

Bibliography

[1] Acompora, Anthony, Srikanth Krishnamurthy, and Scott H. Bloom. "Uninet: A hybrid approach for universal broadband access using small radio cells interconnected by free-space optical links." *IEEE JSAC*, 16(6), August 1998.

[2] Ahuja, Ravindra K, Thomas L. Magnanti, and James B. Orlin. Network Flows: Theory, Algorithms, and Applications. Prentice Hall. Englewood Cliffs, NJ: 1993

[3] Alevras, Dimitris, Martin Groetschel, Peter Jonas, Ulrich Paul, and Roland Wessaely. "Survivable Mobile Phone Network Architectures: Models and Solution Methods." IEEE Communications Magazine. March 1998. pp 88-93.

[4] Bagga, K. S., et al. "Explorations into Graph Vulnerability," *Conference on Graph Theory*, Kalamazoo, Michigan, 1988, pp 29-39.

[5] Bazaraa, Mokhtar S, John J. Jarvis, and Hanif D. Sherali. Linear Programming and Network Flows, Second edition. John Wiley & Sons, Inc. New York: 1990.

[6] Beyer, Dave. "Fundamental Characteristics and Benefits of Wireless Routing in Mesh Networks." Technical report, Wireless Communications Association International, Jan 2002.

[7] Bienstock, D. and E. Gyori. "Average Distance in Graphs with Removed Elements." *Journal of Graph Theory*, Vol 12, 1988, pp 375-390.

[8] Cahn, Robert S. Wide Area Network Design: Concepts and Tools for Optimization. Morgan Kaufmann Publishers, Inc. San Francisco: 1998.

[9] Charddaire, P., et al. "Applications of genetic algorithms in telecommunications." *Proceedings of the Second International Conference on Genetic Algorithms and their Applications*. 257-260, 1987.

[10] Chou, Hsinghua, G. Premkumar, and Chao-Hsien Chu. "Genetic Algorithms and Network Design: An Analysis of Factors Influencing GA's Performance." eBusiness Research Center Working Paper, 09-1999.

[11] Chung, F. R. K. "The Average Distance and the Independence Number." *Journal of Graph Theory*, Vol 12, 1988, pp 229-235.

[12] Coombs, S. and Davis, L. "Genetic algorithms and communication link speed design: constraints and operators." *Proceedings of the Second International Conference on Genetic Algorithms and their Applications*. 257-260, 1987.

[13] Cosares, Steven, and Fred Rispoli. "Module 03-5: Communications Network Design." DIMACS Educational Module Series. DIMACS, October 17, 2003.

[14] Department of Defense. (2000). "Joint Vision 2020. America's Military: Preparing for Tomorrow." Published by: US Government Printing Office, Washington DC, June 2000.

[15] Esbensen, H. "Computing near-optimal solutions to the Steiner Problem in a graph using genetic algorithm." *Networks*. Vol 26, No 4, 173-185, 1995.

[16] Exoo, G. "On a Measure of Communication Network Vulnerability." *Networks*, Vol 12, 1982, pp 405-409.

[17] Farago et al. "VPN Network Topology Optimization Using Random Graphs." INFOCOM, 1999.

[18] Favaron, O., M. Kouider, and M. Makeo. "Edge-Vulnerability and Mean Distance." *Networks*, Vol 19, 1989, pp 493-509.

[18] Fowler, T. "Mesh networks for broadband access." *IEEE Review*, Vol 47, Iss 1:17, January 2001.

[20] Guéret, Christelle, Christian Prins, and Marc Sevaux. "Programmation linéaire." ("Applications of optimization with Xpress-MP." Translated and revised by Susanne Heipcke). Editions Eyrolles. Paris: 2000.

[21] Gupta, P. and P. R. Kumar, "The Capacity of Wireless Networks." *IEEE Transactions on Information Theory*, vol. 46, pp. 388-404, 2000.

[22] Gurumohan, Prabhanjan C. "Topology Design for Free Space Optical Networks." Ph D Qualifying Exam Report for the Department of Electrical Engineering at Arizona State University. 29 July 2003.

[23] Harary, F., F. T. Boesch, and J. A. Kabell. "Graphs as Models of Communication Network Vulnerability: Connectivity and Persistence." *Networks*, Vol 11, 1981, pp 57-63.

[24] Hopkinson, Kenneth and Scott Graham. "Routing in Directional Wireless Networks Using Hybrid Criteria." Submitted to *IEEE Journal on Selected Areas in Communications.*

[25] Hu, Limin. "Topology Control for Multihop Packet Radio Networks." 1991 INFOCOM Conference, Miami, FL.

[26] Izadpanah, Hossein. "A millimiter-wave broadband wireless access technology demonstrator for the next-generation internet network reach extension." *IEEE Communications*, 39(9), September 2001.

[27] Kelleher, James J. "Tactical Communications Network Modeling and Reliability Analysis: An Overview." JSLAI Report JC-2091-GT-F3 under contract DAAL02-89-C-0040, November 1991 (AD-A245339).

[28] Kotsis, Gabriele. "Interconnection Topologies and Routing for Parallel Processing Systems." Technical Report Series, Austrian Center for Parallel Computation, ACPC/TR 92-19, 1992.

[29] Krishnamoorthy, Mohan, Andreas T. Ernst, and Yazid M. Sharaiha. "Comparison of Algorithms for the Degree Constrained Minimum Spanning Tree." *Journal of Heuristics*. 7: 587-611. Nov 2001.

[30] Kruskal, J. B. "On the shortest spanning subtree of a graph and the traveling salesman problem." *Proc. Am. Math. Soc.* Vol. 7, 48-50, 1956.

[31] Maekawa, M. "Optimal processor interconnection topologies." ACM-SIGARCH, May 1981.

[32] McCabe, James D. <u>Network Analysis, Architecture, and Design, Second Edition</u>. Morgan Kaufmann Publishers, New York: 2003.

[33] Narula, S. C. and C. A. Ho. "Degree-constrained minimum spanning tree." *Comput. Op. Res*. Vol 7, 239-249, 1980.

[34] Plesnik, J. "On the Sum of All Distances in a Graph or Digraph." *Journal of Graph Theory*, Vol 8, 1984, pp 1-21.

[35] Pooch, U. W., D. Machuel and J. McCahn. <u>Telecommunications and Networking</u>. CRC Press, Inc. Boca Raton, FL: 1991.

[36] Premkumar, G. and Chao-Hsien Chu. "Telecommunications Network Design Decision: A Genetic Algorithm Approach." eBusiness Research Center Working Paper, 08-1999.

[37] Prim, R. C. "Shortest connection networks and some generalizations." *Bell Sys. Tech. J*. Vol. 36, 1389-1401, 1957.

[38] Savelsbergh, M. and Volgenant, T. "Edge exchanges in the degree-constrained spanning tree problem." *Comput. Op. Res*. Vol 12, 341-348,1985.

[39] Scott, Matthew A, Richard F. Deckro, and James W. Chrissis. "Modeling and Analysis of Multicommodity Network Flows Via Goal Programming." INFOR. Ottawa: May 2005. Vol. 43, Iss. 2; pg 93, 18 pgs.

[40] Tang, Jian, Guolian Xue, and Weiyi Zhang. "Interference-Aware Topology Control and QoS Routing in Muli-Channel Wireless Mesh Networks." *MobiHoc'05*, May 25-28, 2005, Urbana Champaign, Illinois.

[41] Saadawi, T. N., M. H. Ammar and A. E. Hakeem. Fundamentals of Telecommunication Networks. John Wiley & Sons, Inc., New York: 1994.

[42] Shi, Yi, Y. Thomas Hou, Hanif D. Sherali, and Scott F. Midkiff. "Cross-Layer Optimization for Routing Data Traffic in UWB-based Sensor Networks." MobiCom '05. August 28-September 2, 2005, Cologne, Germany.

[43] Von Conta, Christoph. "Torus and other networks as communication networks with up to some hundred points." *IEEE Transactions on Computers*, C-32(7), July 1983

[44] Webb, William. "Broadband fixed wireless access as a key component of the future integrated communications environment." *IEEE Communications*. Vol 39, Iss 9, September 2001.

[45] Winch, R. G. Telecommunication Transmission Systems. McGraw-Hill, New York: 1998.

[46] Xpress-Optimizer Reference Manual. Dash Associates: 2001.

[47] Xue, Q. and A. Ganz. "Ad Hoc QoS On-Demand Routing (AQOR) in Mobile Ad Hoc Networks." *Journal of Parallel Distributed Computing*. Vol 63, 2003, pp 154-165.

[48] Zhao, X., P. B. Luh, and J. Wang. "Surrogate Gradient Algorithm for Lagrangian Relaxation." *Journal of Optimization Theory and Applications*. Vol. 100, No. 3, pp. 699-712, March 1999.

[49] Zhou, Gengui, Mitsuo Gen. "A note in genetic algorithms for degree constrained spanning tree problems." *Networks*. Vol 30, No 2, 91-97, 1997.

[50] Zhou, Gengui and Mitsuo Gen. "Genetic algorithm approach on multi-criteria minimum spanning tree problem." European Journal of Operational Research. v144. 1999.

[51] Zhou, Gengui, Mitsuo Gen, and Tianzu Wu. "A New Approach to the Degree-Constrained Minimum Spanning Tree Problem Using Genetic Algorithm." *IEEE International Conference on Systems, Man, and Cybernetics*. Vol 4 Oct 1996.

Vita

Second Lieutenant Michael C. Erwin graduated from Hoover High School in Hoover, Alabama. He entered undergraduate studies at the United States Air Force Academy in Colorado Springs, Colorado, where he graduated with a Bachelor of Science degree in Mathematical Sciences and received his commission in June 2004. His first assignment was at the Air Force Institute of Technology as a student pursuing a Master's degree in Operations Research. His next assignment is at the Office of Aerospace Studies at Kirtland AFB, NM.

REPORT DOCUMENTATION PAGE	*Form Approved* *OMB No. 074-0188*

1. REPORT DATE *(DD-MM-YYYY)*	**2. REPORT TYPE**	**3. DATES COVERED** *(From – To)*
02-03-2006	**Master's Thesis**	Sep 2004 - Mar 2006

4. TITLE AND SUBTITLE

COMBINING QUALITY OF SERVICE AND TOPOLOGY CONTROL IN DIRECTIONAL HYBRID WIRELESS NETWORKS

5a. CONTRACT NUMBER

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER

6. AUTHOR(S)

Erwin, Michael C., Second Lieutenant, USAF

5d. PROJECT NUMBER

5e. TASK NUMBER

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAMES(S) AND ADDRESS(S)
Air Force Institute of Technology
Graduate School of Engineering and Management (AFIT/EN)
2950 Hobson Street, Building 642
WPAFB OH 45433-7765

8. PERFORMING ORGANIZATION REPORT NUMBER

AFIT/GOR/ENS/06-07

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
AFOSR
Attn: Dr. Robert Herklotz
4015 Wilson Boulevard Room 713
Arlington, VA 22203-1954 Phone: (703) 696-6565

10. SPONSOR/MONITOR'S ACRONYM(S)

11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT
APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED.

13. SUPPLEMENTARY NOTES

14. ABSTRACT
Recent advancements in information and communications technology are changing the information environment in both quantitative and qualitative measures. The developments in directional wireless capabilities necessitate the ability to model these new capabilities, especially in dynamic environments typical of military combat operations. This thesis establishes a foundation for the definition and consideration of the unique network characteristics and requirements introduced by this novel instance of the Network Design Problem (NDP). Developed are a Mixed-Integer Linear Program (MILP) formulation and two heuristic strategies for solving the NDP. A third solution strategy using the MILP formulation with a degree-constrained Minimum Spanning Tree starting solution is also considered. The performance of the various methods are evaluated on the basis of solution quality, computation time, and other network metrics via randomly generated data sets for several different problem sizes.

15. SUBJECT TERMS
Quality of Service, Topology Control, Directional Hybrid Wireless Networks, Network Design Problem, Communications Synthesis, Telecommunications Networks

16. SECURITY CLASSIFICATION OF:			**17. LIMITATION OF ABSTRACT**	**18. NUMBER OF PAGES**	**19a. NAME OF RESPONSIBLE PERSON** Raymond W Staats, Lt Col, USAF (ENS)
a. REPORT	**b. ABSTRACT**	**c. THIS PAGE**			**19b. TELEPHONE NUMBER** *(Include area code)*
U	**U**	**U**	**UU**	117	(937) 785-3636, ext 4624; e-mail: Raymond.Staats@afit.edu

Standard Form 298 (Rev. 8-98)
Prescribed by ANSI Std. Z39-18